

Documented Code For glossaries v4.03

Nicola L.C. Talbot

Dickimaw Books

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

2014-01-20

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 v4.03: \TeX 2e 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.

Contents

1 Main Package Code	3
1.1 Package Definition	3
1.2 Package Options	5
1.3 Default values	26
1.4 Xindy	36
1.5 Loops and conditionals	45
1.6 Defining new glossaries	50
1.7 Defining new entries	53
1.8 Resetting and unsetting entry flags	73
1.9 Loading files containing glossary entries	75
1.10 Using glossary entries in the text	76
1.10.1 Links to glossary entries	86
1.10.2 Displaying entry details without adding information to the glossary	132
1.11 Adding an entry to the glossary without generating text	139
1.12 Creating associated files	140
1.13 Writing information to associated files	150
1.14 Glossary Entry Cross-References	155
1.15 Displaying the glossary	156
1.16 Acronyms	173
1.17 Predefined acronym styles	179
1.18 Predefined Glossary Styles	210
1.19 Debugging Commands	210
1.20 Compatibility with version 2.07 and below	215
2 Prefix Support (glossaries-prefix Code)	216
3 Mfirstuc Documented Code	222
4 Glossary Styles	225
4.1 Glossary hyper-navigation definitions (glossary-hypernav package)	225
4.2 In-line Style (glossary-inline.sty)	227
4.3 List Style (glossary-list.sty)	229
4.4 Glossary Styles using longtable (the glossary-long package)	233
4.5 Glossary Styles using longtable (the glossary-longragged package)	239
4.6 Glossary Styles using multicol (glossary-mcols.sty)	244
4.7 Glossary Styles using supertabular environment (glossary-super package)	248
4.8 Glossary Styles using supertabular environment (glossary-superragged package)	255
4.9 Tree Styles (glossary-tree.sty)	260
5 glossaries-compatible-207	268

6 Accessibility Support (glossaries-accsupp Code)	288
6.1 Defining Replacement Text	289
6.2 Accessing Replacement Text	292
6.3 Displaying the Glossary	308
6.4 Acronyms	309
6.5 Debugging Commands	323
7 Multi-Lingual Support	325
7.1 Babel Captions	325
7.2 Polyglossia Captions	331
7.3 Brazilian Dictionary	334
7.4 Danish Dictionary	334
7.5 Dutch Dictionary	334
7.6 English Dictionary	335
7.7 French Dictionary	335
7.8 German Dictionary	335
7.9 Irish Dictionary	336
7.10 Italian Dictionary	336
7.11 Magyar Dictionary	336
7.12 Polish Dictionary	337
7.13 Serbian Dictionary	337
7.14 Spanish Dictionary	337
Glossary	337
Change History	338
Index	356

1 Main Package Code

1.1 Package Definition

This package requires $\text{\LaTeX} 2_{\epsilon}$.

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2014/01/20 v4.03 (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.

```
\glsorg@glossary First, save the original behaviour of \glossary
```

```
21 \newcommand{\glsorg@glossary}{%
```

```
22   \@bsphack
```

```
23   \begingroup
```

```
24     \@sanitize \endgroup\@esphack
```

```
25 }
```

```
\glsorg@wrglossary
```

```
26 \newcommand{\glsorg@wrglossary}[1]{%
```

```
27   \protected@write\@glossaryfile{}{%
```

```
28     \string \glossaryentry{#1}{\thepage}}%
```

```
29   \endgroup
```

```
30   \@esphack
```

```
31 }
```

```
32 \renewcommand*{\RecordChanges}{%
```

```
33   \newwrite\@glossaryfile
```

```
34   \immediate\openout\@glossaryfile=\jobname.glo
```

```
35   \def\glsorg@glossary{\@bsphack\begingroup\@sanitize\glsorg@wrglossary}%
```

```
36   \typeout{Writing glossary file \jobname .glo}%
```

```
37 }
```

`\changes` Now we need to redefine `\changes` so that it uses the original definition of `\glossary`.

```

38 \let\glsorg@changes\changes
39 \renewcommand{\changes}[3]{%
40   \begingroup
41     \let\glossary\glsorg@glossary
42     \glsorg@changes{#1}{#2}{#3}%
43   \endgroup
44 }
```

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

`\glsorg@theglossary`

```
45 \let\glsorg@theglossary\theglossary
```

`\glsorg@endtheglossary`

```
46 \let\glsorg@endtheglossary\endtheglossary
```

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

```

47 \let\glsorg@PrintChanges\PrintChanges
48 \renewcommand{\PrintChanges}{%
49   \begingroup
50     \let\theglossary\glsorg@theglossary
51     \let\endtheglossary\glsorg@endtheglossary
52     \glsorg@PrintChanges
53   \endgroup
54 }
```

End of doc stuff.

```
55 \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.

```
56 \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`.

```
57 \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`.

```

58 \ifcsundef{chapter}%
59   {\newcommand*\@@glossarysec{section}}%
60   {\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 redefine `\glossarysection`.

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

Determine whether or not to use numbered sections.

`\@@glossarysecstar`

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

`\@@glossaryseclabel`

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

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

```
66 \newcommand*{\glsautoprefix}{}
```

`numberedsection`

```
67 \define@choicekey{glossaries.sty}{numberedsection}[\val\nr]{%
68 false,nolabel,autolabel,nameref}[nolabel]{%
69   \ifcase\nr\relax
70     \renewcommand*{\@@glossarysecstar}{*}%
71     \renewcommand*{\@@glossaryseclabel}{}%
72   \or
73     \renewcommand*{\@@glossarysecstar}{}%
74     \renewcommand*{\@@glossaryseclabel}{}%
75   \or
76     \renewcommand*{\@@glossarysecstar}{}%
77     \renewcommand*{\@@glossaryseclabel}{}%
78     \label{\glsautoprefix@glo@type}%
79   \or
80     \renewcommand*{\@@glossarysecstar}{*}%
81     \renewcommand*{\@@glossaryseclabel}{}%
82     \protected@edef\@currentlabelname{\glossarytoctitle}%
83     \label{\glsautoprefix@glo@type}%
84   \fi
85 }
```

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`

```
86 \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](#).

```
87 \define@key{glossaries.sty}{style}{%
88   \renewcommand*{\@glossary@default@style}{#1}%
89 }
```

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`

```
90 \newcommand*{\@gls@declareoption}[2]{%
91   \DeclareOptionX{#1}{#2}%
92   \DeclareOption{#1}{#2}%
93 }
```

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`

```
94 \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).

```
95 \@gls@declareoption{nonumberlist}{%
96   \renewcommand*{\glossaryentrynumbers}[1]{\gls@save@numberlist{#1}}%
97 }
```

savenumberlist Provide means to store the number list for entries.

```
98 \define@boolkey{glossaries.sty}[gls]{savenumberlist}[true]{}
99 \glssavenumberlistfalse
```

`\@glo@seeautonumberlist`

```
100 \newcommand*{\@glo@seeautonumberlist{}}
```

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

```
101 \@gls@declareoption{seeautonumberlist}{%
102   \renewcommand*{\@glo@seeautonumberlist}{%
103     \def\@glo@prefix{\glsnextpages}%
104   }%
105 }
```

```

\@gls@loadlong
106 \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 pro-
vided to reduce overhead caused by loading unrequired packages.
107 \@gls@declareoption{nolong}{\renewcommand*{\@gls@loadlong}{}}

\@gls@loadsuper The package isn't loaded if isn't installed.
108 \IfFileExists{supertabular.sty}{%
109 \newcommand*{\@gls@loadsuper}{\RequirePackage{glossary-super}}}{%
110 \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 pro-
vided to reduce overhead caused by loading unrequired packages.
111 \@gls@declareoption{nosuper}{\renewcommand*{\@gls@loadsuper}{}}

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

nolist This option prevents from being loaded (to reduce overheads if required). Nat-
urally, the styles defined in will not be available if this option is used.
113 \@gls@declareoption{nolist}{\renewcommand*{\@gls@loadlist}{}}

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

notree This option prevents from being loaded (to reduce overheads if required). Nat-
urally, the styles defined in will not be available if this option is used.
115 \@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).
116 \@gls@declareoption{nostyles}{%
117 \renewcommand*{\@gls@loadlong}{}%
118 \renewcommand*{\@gls@loadsuper}{}%
119 \renewcommand*{\@gls@loadlist}{}%
120 \renewcommand*{\@gls@loadtree}{}%
121 \let\@glossary@default@style\relax
122 }

\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 ad-
justed in case the description ends with an upper case letter. (Patch provided
by Michael Pock.)

```



```

123 \newcommand*{\glspostdescription}{%
124   \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi
125 }

```

nopostdot Boolean option to suppress post description dot

```

126 \define@boolkey{glossaries.sty}[gls]{nopostdot}[true]{}
127 \glsnopostdotfalse

```

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

```

128 \define@boolkey{glossaries.sty}[gls]{nogroupskip}[true]{}
129 \glsnogroupskipfalse

```

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

```

130 \define@boolkey{glossaries.sty}[gls]{ucmark}[true]{}

131 \@ifclassloaded{memoir}
132 {%
133   \glsucmarktrue
134 }%
135 {%
136   \glsucmarkfalse
137 }

```

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`.

```

138 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
139 \glsentrycounterfalse

```

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

```

140 \define@key{glossaries.sty}{counterwithin}{%
141   \renewcommand*{\@gls@counterwithin}{#1}%
142   \glsentrycountertrue
143 }

```

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

```

144 \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`.

```

145 \define@boolkey{glossaries.sty}[gls]{subentrycounter}[true]{}
146 \glssubentrycounterfalse

```

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

```

147 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
148   \csname @gls@setupsort@#1\endcsname
149 }

```

`\glsprestandardsort` `\glsprestandardsort{<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.

```

150 \newcommand*{\glsprestandardsort}[3]{%
151   \glsdosanitizesort
152 }

```

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

```

153 \newcommand*{\@gls@setupsort@standard}{%

```

Store entry information when it's defined.

```

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

```

No count register required for standard sort.

```

155   \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.)

```

156   \def\@gls@defsort##1##2{%

```

```

157     \ifx\@glo@sort\@glsdefaultsort

```

```

158       \let\@glo@sort\@glo@name

```

```

159     \fi

```

```

160     \let\glsdosanitizesort\@gls@sanitizesort

```

```

161     \glsprestandardsort{\@glo@sort}{##1}{##2}%

```

```

162     \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%

```

```

163   }%

```

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

```

164   \def\@gls@setsort##1{%

```

```

165 }

```

Set standard sort as the default:

```

166 \@gls@setupsort@standard

```

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

```

167 \newcommand*{\glssortnumberfmt}[1]{%

```

```

168   \ifnum#1<100000 0\fi

```

```

169   \ifnum#1<10000 0\fi

```

```

170   \ifnum#1<1000 0\fi

```

```

171   \ifnum#1<100 0\fi

```

```

172 \ifnum#1<10 0\fi
173 \number#1%
174 }

\@gls@setupsort@def Set up the macros for order of definition sorting.
175 \newcommand*{\@gls@setupsort@def}{%
    Store entry information when it's defined.
176 \def\do@glo@storeentry{\@glo@storeentry}%
    Defined count register associated with the glossary.
177 \def\@gls@defsortcount##1{%
178 \expandafter\global
179 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
180 }%
    Increment count register associated with the glossary and use as the sort key.
181 \def\@gls@defsort##1##2{%
182 \expandafter\global\expandafter
183 \advance\csname glossary@##1@sortcount\endcsname by 1\relax
184 \expandafter\protected@xdef\csname glo@##2@sort\endcsname{%
185 \expandafter\glssortnumberfmt
186 {\csname glossary@##1@sortcount\endcsname}}%
187 }%
    Don't need to do anything when the entry is used.
188 \def\@gls@setsort##1{%
189 }

\@gls@setupsort@use Set up the macros for order of use sorting.
190 \newcommand*{\@gls@setupsort@use}{%
    Don't store entry information when it's defined.
191 \let\do@glo@storeentry\@gobble
    Defined count register associated with the glossary.
192 \def\@gls@defsortcount##1{%
193 \expandafter\global
194 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
195 }%
    Initialise the sort key to empty.
196 \def\@gls@defsort##1##2{%
197 \expandafter\gdef\csname glo@##2@sort\endcsname{}%
198 }%
    If the sort key hasn't been set, increment the counter associated with the glossary and set the sort key.
199 \def\@gls@setsort##1{%
    Get the parent, if one exists
200 \edef\@glo@parent{\csname glo@##1@parent\endcsname}%

```

Set the information for the parent entry if not already done.

```

201 \ifx\@glo@parent\@empty
202 \else
203 \expandafter\@gls@setsort\expandafter{\@glo@parent}%
204 \fi
Set index information for this entry
205 \edef\@glo@type{\csname glo@##1@type\endcsname}%
206 \edef\@gls@tmp{\csname glo@##1@sort\endcsname}%
207 \ifx\@gls@tmp\@empty
208 \expandafter\global\expandafter
209 \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
210 \expandafter\protected@xdef\csname glo@##1@sort\endcsname{%
211 \expandafter\glssortnumberfmt
212 {\csname glossary@\@glo@type @sortcount\endcsname}}%
213 \@glo@storeentry{##1}%
214 \fi
215 }%
216 }

```

`\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.)

```

217 \newcommand*\glsdefmain{%
218 \if@gls@docloaded
219 \newglossary[glg2]{main}{gls2}{glo2}{\glossaryname}%
220 \else
221 \newglossary{main}{gls}{glo}{\glossaryname}%
222 \fi
223 }

```

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`

```

224 \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`

```

225 \newcommand*\acronymtype{\glsdefaulttype}

```

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

```
226 \@gls@declareoption{nomain}{%
227   \let\glsdefaulttype\relax
228   \renewcommand*{\glsdefmain}{}%
229 }
```

`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.

```
230 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
231   \ifglsacronym
232     \renewcommand{\@gls@do@acronymsdef}{%
233       \DeclareAcronymList{acronym}%
234       \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
235       \renewcommand*{\acronymtype}{acronym}%
236     }%
237   \else
238     \let\@gls@do@acronymsdef\relax
239   \fi
240 }
```

`\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.

```
241 \AtBeginDocument{%
242   \ifglsacronym
243     \ifbool{glscompatible-3.07}%
244     {%
245     }%
246     \providecommand*{\printacronyms}[1][1]{%
247       \printglossary[type=\acronymtype,#1]}%
248   }%
249   \fi
250 }
```

`@gls@do@acronymsdef` Set default value

```
251 \newcommand*{\@gls@do@acronymsdef}{}
```

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

```
252 \@gls@declareoption{acronyms}{%
253   \glsacronymtrue
254   \renewcommand{\@gls@do@acronymsdef}{%
255     \DeclareAcronymList{acronym}%
256     \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
257     \renewcommand*{\acronymtype}{acronym}%
258   }%
259 }
```

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

```
260 \newcommand*{\@glsacronymlists}{}
```

`\@addtoacronymlists`

```
261 \newcommand*{\@addtoacronymlists}[1]{%
262   \ifx\@glsacronymlists\@empty
263     \protected@xdef\@glsacronymlists{#1}%
264   \else
265     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
266   \fi
267 }
```

`\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.)

```
268 \newcommand*{\DeclareAcronymList}[1]{%
269   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
270 }
```

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

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

```
271 \newcommand{\glsIfListOfAcronyms}[1]{%
272   \edef\@do@gls@islistofacronyms{%
273     \noexpand\@gls@islistofacronyms{#1}{\@glsacronymlists}}%
274   \@do@gls@islistofacronyms
275 }
```

Internal command requires label and list to be expanded:

```
276 \newcommand{\@gls@islistofacronyms}[4]{%
277   \def\@gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
278     \def\@before{##1}\def\@after{##2}}%
279   \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
280   \ifx\@after\@nnil
```

Not found

```
281   #4%
282   \else
```

Found

```
283   #3%
284   \fi
285 }
```

`if@glsisacronymlist` Convenient boolean.

```

286 \newif\if@glsisacronymlist

```

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

```

287 \newcommand*{\gls@checkisacronymlist}[1]{%
288   \glsIfListOfAcronyms{#1}%
289   {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
290 }

```

`\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.)

```

291 \newcommand*{\SetAcronymLists}[1]{%
292   \renewcommand*{\@glsacronymlists}{#1}%
293 }

```

`acronymlists`

```

294 \define@key{glossaries.sty}{acronymlists}{%
295   \DeclareAcronymList{#1}%
296 }

```

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`

```

297 \newcommand{\glscounter}{page}

```

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

```

298 \define@key{glossaries.sty}{counter}{%
299   \renewcommand*{\glscounter}{#1}%
300 }

```

`\@gls@nohyperlist`

```

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

```

`sDeclareNoHyperList`

```

302 \newcommand*{\GlsDeclareNoHyperList}[1]{%
303   \ifdefempty\@gls@nohyperlist
304   {%
305     \renewcommand*{\@gls@nohyperlist}{#1}%
306   }%
307   {%
308     \appto\@gls@nohyperlist{,#1}%
309   }%
310 }

```

nohypertypes

```
311 \define@key{glossaries.sty}{nohypertypes}{%
312   \GlsDeclareNoHyperList{#1}%
313 }
```

\GlossariesWarning Prints a warning message.

```
314 \newcommand*{\GlossariesWarning}[1]{%
315   \PackageWarning{glossaries}{#1}%
316 }
```

sariesWarningNoLine Prints a warning message without the line number.

```
317 \newcommand*{\GlossariesWarningNoLine}[1]{%
318   \PackageWarningNoLine{glossaries}{#1}%
319 }
```

nowarn Define package option to suppress warnings

```
320 \@gls@declareoption{nowarn}{%
321   \renewcommand*{\GlossariesWarning}[1]{}%
322   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
323 }
```

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

```
324 \newcommand*{\@gls@sanitizedesc}{%
325 }
326 %\end{macro}
327 %
328 %\begin{macro}{\glssetexpandfield}
329 %\changes{3.13a}{2013-11-05}{new}
330 %\begin{definition}
331 %\cs{glssetexpandfield}\marg{field}
332 %\end{definition}
333 % Sets field to always expand.
334 %   \begin{macrocode}
335 \newcommand*{\glssetexpandfield}[1]{%
336   \csdef{gls@assign@#1@field}##1##2{%
337     \@gls@expand@field{##1}{#1}{##2}%
338   }%
339 }
```

\glssetnoexpandfield

\glssetnoexpandfield{<field>}

Sets field to never expand.


```

340 \newcommand*{\glsetnoexpandfield}[1]{%
341   \csdef{gls@assign@#1@field}##1##2{%
342     \@@gls@noexpand@field{##1}{#1}{##2}%
343   }%
344 }

```

s@assign@type@field The type must always be expandable.

```
345 \glsetexpandfield{type}
```

s@assign@desc@field The description is not expanded by default:

```
346 \glsetnoexpandfield{desc}
```

gn@descplural@field

```
347 \glsetnoexpandfield{descplural}
```

\@gls@sanitizename

```
348 \newcommand*{\@gls@sanitizename}{}
```

s@assign@name@field Don't expand name by default.

```
349 \glsetnoexpandfield{name}
```

@gls@sanitizesymbol

```
350 \newcommand*{\@gls@sanitizesymbol}{}
```

assign@symbol@field Don't expand symbol by default.

```
351 \glsetnoexpandfield{symbol}
```

@symbolplural@field

```
352 \glsetnoexpandfield{symbolplural}
```

Sanitizing stuff:

\@gls@sanitizesort

```

353 \newcommand*{\@gls@sanitizesort}{%
354   \ifglssanitizesort
355     \@onelevel@sanitize\@glo@sort
356   \else
357     \fi
358 }

```

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.

```

359 \define@boolkey[gls]{sanitize}{description}[true]{%
360   \GlossariesWarning{sanitize={description} package option deprecated}%
361   \ifgls@sanitize@description
362     \glsetnoexpandfield{desc}%
363     \glsetnoexpandfield{descplural}%

```

```

364 \else
365   \glssetexpandfield{desc}%
366   \glssetexpandfield{descplural}%
367 \fi
368 }

369 \define@boolkey[glS]{sanitize}{name}[true]{%
370   \GlossariesWarning{sanitize={name} package option deprecated}%
371   \ifglS@sanitize@name
372     \glssetnoexpandfield{name}%
373   \else
374     \glssetexpandfield{name}%
375   \fi
376 }

377 \define@boolkey[glS]{sanitize}{symbol}[true]{%
378   \GlossariesWarning{sanitize={symbol} package option deprecated}%
379   \ifglS@sanitize@symbol
380     \glssetnoexpandfield{symbol}%
381     \glssetnoexpandfield{symbolplural}%
382   \else
383     \glssetexpandfield{symbol}%
384     \glssetexpandfield{symbolplural}%
385   \fi
386 }

```

sanitizesort

```

387 \define@boolkey{glossaries.sty}[glS]{sanitizesort}[true]{%
388   \ifglssanitizesort
389     \glssetnoexpandfield{sortvalue}%
390   \else
391     \glssetexpandfield{sortvalue}%
392   \fi
393 }

```

Default setting:

```

394 \glssanitizesorttrue
395 \glssetnoexpandfield{sortvalue}%

396 \define@choicekey[glS]{sanitize}{sort}{true,false}[true]{%
397   \setbool{glssanitizesort}{#1}%
398   \ifglssanitizesort
399     \glssetnoexpandfield{sortvalue}%
400   \else
401     \glssetexpandfield{sortvalue}%
402   \fi
403   \GlossariesWarning{sanitize={sort} package option
404     deprecated. Use sanitizesort instead}%
405 }

```

sanitize

```

406 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
407 name=true]{%
408   \ifthenelse{\equal{#1}{none}}{%
409     {%
410       \GlossariesWarning{sanitize package option deprecated}%
411     }%
412   }%
413   \setkeys[glS]{sanitize}{#1}%
414 }%
415 }

```

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

```

416 \newif\ifglstranslate

```

`ls@nottranslatorhook`

```

417 \newcommand*{\@glS@nottranslatorhook}{

```

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

```

418 \@glS@declareoption{nottranslate}{%
419   \glstranslatefalse
420   \let\@glS@nottranslatorhook\relax
421 }

```

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

```

422 \define@choicekey{glossaries.sty}{translate}[\val\nr]%
423 {true,false,babel}[true]%
424 {%
425   \ifcase\nr\relax
426     \glstranslatetrue
427   \or
428     \glstranslatefalse
429     \let\@glS@nottranslatorhook\relax
430   \or
431     \glstranslatefalse
432     \def\@glS@nottranslatorhook{\RequirePackage{glossaries-babel}}%
433   \fi
434 }

```

Set the default value:

```

435 \glstranslatefalse
436 \@ifpackageloaded{translator}%
437   {\glstranslatetrue}%
438   {%
439     \@ifpackageloaded{polyglossia}%
440       {\glstranslatetrue}%
441       {%

```

```

442         \@ifpackageloaded{babel}{\glstranslatetrue}{}%
443     }%
444 }

```

indexonlyfirst Set whether to only index on first use.

```

445 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
446 \glsindexonlyfirstfalse

```

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

```

447 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{}
448 \glshyperfirsttrue

```

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

```

449 \newcommand*{\@gls@setacrstyle}{}

```

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

```

450 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{}%
451 \ifbool{glsacrdescription}%
452 {}%
453 {}%
454 \renewcommand*{\@gls@sanitizedesc}{}%
455 }%
456 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
457 }

```

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

```

458 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{}%
459 \renewcommand*{\@gls@sanitizesymbol}{}%
460 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
461 }

```

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

```

462 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{}%
463 \renewcommand*{\@gls@sanitizesymbol}{}%
464 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
465 }

```

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

```

466 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{}%
467 \renewcommand*{\@gls@sanitizesymbol}{}%
468 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
469 }

```

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

```

470 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
471   \renewcommand*{\@gls@sanitizesymbol}{}%
472   \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
473 }

```

shortcuts Define acronym shortcuts.

```

474 \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.

```

475 \newcommand*{\glsorder}{word}

```

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

```

476 \newcommand*{\@glsorder}[1]{}

```

order

```

477 \define@choicekey{glossaries.sty}{order}{word,letter}{%
478   \def\glsorder{#1}}

```

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

```

479 \newif\ifglxindy

```

The default is makeindex:

```

480 \glxindyfalse

```

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

```

481 \@gls@declareoption{makeindex}{\glxindyfalse}

```

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.

```

482 \define@boolkey[gls]{xindy}{glsnumbers}[true]{}
483 \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.)

```

484 \def\@xdy@main@language{\language}%

```

Define key to set the language

```

485 \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.

```
486 \ifcsundef{inputencodingname}{%
487   \def\gls@codepage{}}{%
488   \def\gls@codepage{\inputencodingname}
489 }
```

Define a key to set the code page.

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

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

```
491 \define@key{glossaries.sty}{xindy}[]{%
492   \glsxindytrue
493   \setkeys[gls]{xindy}{#1}%
494 }
```

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

```
495 \@gls@declareoption{xindygloss}{%
496   \glsxindytrue
497 }
```

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

```
498 \@gls@declareoption{xindynoglsnumbers}{%
499   \glsxindytrue
500   \gls@xindy@glsnumbersfalse
501 }
```

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

```
502 \define@boolkey{glossaries.sty}[gls]{savewrites}[true]{%
503   \ifglssavewrites
504     \renewcommand*{\glswritefiles}{\@glswritefiles}%
505   \else
506     \let\glswritefiles\@empty
507   \fi
508 }
```

Set default:

```
509 \glssavewritesfalse
510 \let\glswritefiles\@empty
```

`compatible-3.07`

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

`compatible-2.07`

```
513 \define@boolkey{glossaries.sty}[gls]{compatible-2.07}[true]{%
```

Also set 3.07 compatibility if this option is set.

```

514 \ifbool{glscompatible-2.07}%
515 {%
516   \booltrue{glscompatible-3.07}%
517 }%
518 {}%
519 }
520 \boolfalse{glscompatible-2.07}

```

symbols Create a “symbols” glossary type

```

521 \@gls@declareoption{symbols}{%
522   \let\@gls@do@symbolsdef\@gls@symbolsdef
523 }

```

Default is not to define the symbols glossary:

```

524 \newcommand*{\@gls@do@symbolsdef}{}

```

\@gls@symbolsdef

```

525 \newcommand*{\@gls@symbolsdef}{%
526   \newglossary[slg]{symbols}{sls}{slo}{\glssymbolsgroupname}%
527   \newcommand*{\printsymbols}[1][]{\printglossary[type=symbols,##1]}%
528 }%

```

numbers Create a “symbols” glossary type

```

529 \@gls@declareoption{numbers}{%
530   \let\@gls@do@numbersdef\@gls@numbersdef
531 }

```

Default is not to define the numbers glossary:

```

532 \newcommand*{\@gls@do@numbersdef}{}

```

\@gls@numbersdef

```

533 \newcommand*{\@gls@numbersdef}{%
534   \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%
535   \newcommand*{\printnumbers}[1][]{\printglossary[type=numbers,##1]}%
536 }%

```

index Create an “index” glossary type

```

537 \@gls@declareoption{index}{%
538   \let\@gls@do@indexdef\@gls@indexdef
539 }

```

Default is not to define index glossary:

```

540 \newcommand*{\@gls@do@indexdef}{}

```

\@gls@indexdef

```

541 \newcommand*{\@gls@indexdef}{%
542   \newglossary[ilg]{index}{ind}{idx}{\indexname}%
543   \newcommand*{\printindex}[1][]{\printglossary[type=index,##1]}%

```

```

544 \newcommand*{\newterm}[2] [] {%
545   \newglossaryentry{##2}%
546   {type={index},name={##2},description={\nopostdesc},##1}}
547 }%

```

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

```

548 \@for\CurrentOption :=\@declaredoptions\do{%
549   \ifx\CurrentOption\@empty
550   \else
551     \@expandtwoargs
552     \in@ {,\CurrentOption ,}{,\@classoptionslist,\@curroptions,}%
553     \ifin@
554     \@use@option
555     \expandafter \let\csname ds@\CurrentOption\endcsname\@empty
556     \fi
557   \fi
558 }

```

Now process options passed to the package:

```

559 \ProcessOptionsX

```

Load backward compatibility stuff:

```

560 \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:

```

561 \disable@keys{glossaries.sty}{compatible-2.07,%
562 xindy,xindygloss,xindynoglsnumbers,makeindex,%
563 acronym,translate,nottranslate,nolong,nosuper,notree,nostyles,nomain}

```

Now define `\setupglossaries`:

```

564 \newcommand*{\setupglossaries}[1] {%
565   \renewcommand*{\@gls@setacrstyle}{}%
566   \ifglsacrshortcuts
567     \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
568   \else
569     \def\@gls@setupshortcuts{%
570       \ifglsacrshortcuts
571         \DefineAcronymSynonyms
572       \fi
573     }%
574   \fi
575   \glsacrshortcutsfalse
576   \let\@gls@do@numbersdef\relax
577   \let\@gls@do@symbolssdef\relax
578   \let\@gls@do@indexdef\relax
579   \let\@gls@do@acronymsdef\relax
580   \setkeys{glossaries.sty}{#1}%
581   \@gls@setacrstyle

```



```

582 \@gls@setupshortcuts
583 \@gls@do@acronymsdef
584 \@gls@do@numbersdef
585 \@gls@do@symbolssdef
586 \@gls@do@indexdef
587 }

```

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

```

588 \ifglstranslate
589 \@ifpackageloaded{polyglossia}%
590 {%
    polyglossia fakes babel so need to check for polyglossia first.
591 }%
592 {%
593     \@ifpackageloaded{babel}%
594     {%
595         \IfFileExists{translator.sty}%
596         {%
597             \RequirePackage{translator}%
598         }%
599     }%
600 }%
601 {}
602 }
603 \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.<n>.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 `{<section-level>.<n>.0}` non-existent warning (e.g. `\gls[counter=chapter]{label}`).

```

604 \ifthenelse{\equal{\glscounter}{section}}{%
605 {%
606     \ifcsundef{chapter}{}%
607     {%
608         \let\@gls@old@chapter\@chapter
609         \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
610             \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}}%
611     }%
612 }%
613 {}

```

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

```
614 \newcommand*{\@gls@onlypremakeg}{}
```

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

```
615 \newcommand*{\@onlypremakeg}[1]{%
616   \ifx\@gls@onlypremakeg\@empty
617     \def\@gls@onlypremakeg{#1}%
618   \else
619     \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
620     \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
621   \fi
622 }
```

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

```
623 \newcommand*{\@disable@onlypremakeg}{%
624   \@for\@thiscs:=\@gls@onlypremakeg\do{%
625     \expandafter\@disable@premakecs\@thiscs%
626   }}
```

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

```
627 \newcommand*{\@disable@premakecs}[1]{%
628   \def#1{\PackageError{glossaries}{\string#1\space may only be
629     used before \string\makeglossaries}{You can't use
630     \string#1\space after \string\makeglossaries}}%
631 }
```

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`

```
632 \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`

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

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

```
634 \newcommand*{\glssettoctitle}[1]{%
635   \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`
636 `\providecommand*{\entryname}{Notation}`

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

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

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

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

`glssymbolsgroupname`
640 `\providecommand*{\glssymbolsgroupname}{Symbols}`

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

`\glspluralsuffix` The default plural is formed by appending `\glspluralsuffix` to the singular form.
642 `\newcommand*{\glspluralsuffix}{s}`

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

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

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

`dglossarytocaptions` 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.)
645 `\newcommand*{\addglossarytocaptions}[1]{%`
646 `\ifcsundef{captions#1}{}%`
647 `{%`
648 `\expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname`
649 `\expandafter\toks@\expandafter{\@gls@tmp`
650 `\renewcommand*{\glossaryname}{\translate{Glossary}}%`
651 `}%`
652 `\expandafter\edef\csname captions#1\endcsname{\the\toks@}%`
653 `}%`
654 `}`

655 \ifglstranslate

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

```
656 \@ifpackageloaded{translator}{%
657   \usedictionary{glossaries-dictionary}%
658   \addglossarytocaptions{portuges}%
659   \addglossarytocaptions{portuguese}%
660   \addglossarytocaptions{brazil}%
661   \addglossarytocaptions{brazilian}%
662   \addglossarytocaptions{danish}%
663   \addglossarytocaptions{dutch}%
664   \addglossarytocaptions{afrikaans}%
665   \addglossarytocaptions{english}%
666   \addglossarytocaptions{UKenglish}%
667   \addglossarytocaptions{USenglish}%
668   \addglossarytocaptions{american}%
669   \addglossarytocaptions{australian}%
670   \addglossarytocaptions{british}%
671   \addglossarytocaptions{canadian}%
672   \addglossarytocaptions{newzealand}%
673   \addglossarytocaptions{french}%
674   \addglossarytocaptions{frenchb}%
675   \addglossarytocaptions{francais}%
676   \addglossarytocaptions{acadian}%
677   \addglossarytocaptions{canadien}%
678   \addglossarytocaptions{german}%
679   \addglossarytocaptions{germanb}%
680   \addglossarytocaptions{austrian}%
681   \addglossarytocaptions{naustrian}%
682   \addglossarytocaptions{ngerman}%
683   \addglossarytocaptions{irish}%
684   \addglossarytocaptions{italian}%
685   \addglossarytocaptions{magyar}%
686   \addglossarytocaptions{hungarian}%
687   \addglossarytocaptions{polish}%
688   \addglossarytocaptions{spanish}%
689   \renewcommand*{\glstttitle}[1]{%
690     \ifthenelse{\equal{#1}{main}}{%
691       \translatelet{\glossarytoctitle}{Glossary}}{%
692       \ifthenelse{\equal{#1}{acronym}}{%
693         \translatelet{\glossarytoctitle}{Acronyms}}{%
694         \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
695   \renewcommand*{\glossaryname}{\translate{Glossary}}%
696   \renewcommand*{\acronymname}{\translate{Acronyms}}%
697   \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
698   \renewcommand*{\descriptionname}{%
699     \translate{Description (glossaries)}}%
700   \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
701   \renewcommand*{\pagelistname}{%
702     \translate{Page List (glossaries)}}%
```

```

703   \renewcommand*{\glssymbolsgroupname}{%
704     \translate{Symbols (glossaries)}}%
705   \renewcommand*{\glsnumbersgroupname}{%
706     \translate{Numbers (glossaries)}}%
707   }{%

708   \@ifpackageloaded{polyglossia}%
709   {\RequirePackage{glossaries-polyglossia}}%
710   {%
711     \@ifpackageloaded{babel}{%
712       \RequirePackage{glossaries-babel}}}%
713   }}
714 \else

715   \@gls@notranslatorhook
716 \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.

```
717 \DeclareRobustCommand*{\nopostdesc}{}

```

`\@nopostdesc` Suppress next description terminator.

```

718 \newcommand*{\@nopostdesc}{%
719   \let\org@glspostdescription\glspostdescription
720   \def\glspostdescription{%
721     \let\glspostdescription\org@glspostdescription}%
722 }

```

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

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

```

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

```
724 \newcommand{\glspar}{\par}

```

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

```

725 \ifglxsindy
726   \newcommand{\setStyleFile}[1]{%
727     \renewcommand{\istfilename}{#1.xdy}}
728 \else
729   \newcommand{\setStyleFile}[1]{%
730     \renewcommand{\istfilename}{#1.ist}}
731 \fi

```

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

```
732 \@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 re-defining 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`

```
733 \ifglxindy
734   \def\istfilename{\jobname.xdy}
735 \else
736   \def\istfilename{\jobname.ist}
737 \fi
```

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 \TeX , `\@istfilename` ignores its argument.

`\@istfilename`

```
738 \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`

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

`\glsSetCompositor` Sets the compositor.

```
740 \newcommand*\glsSetCompositor[1]{%
741   \renewcommand*\glscompositor{#1}}
Only use before \makeglossaries
742 \@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 `\langle letter \rangle \langle compositor \rangle \langle number \rangle`. 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`.

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

`\sSetAlphaCompositor` Sets the alpha compositor.

```
744 \ifglxindy
745   \newcommand*\glsSetAlphaCompositor[1]{%
746     \renewcommand*\@glsAlphacompositor{#1}}
747 \else
748   \newcommand*\glsSetAlphaCompositor[1]{%
749     \glsnoxindywarning\glsSetAlphaCompositor}
750 \fi
```

Can only be used before `\makeglossaries`

```
751 \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.

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

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

```
753 \newcommand*\glsSetSuffixF[1]{%
```

```
754 \renewcommand*\gls@suffixF{#1}}
```

Only has an effect when used before `\makeglossaries`

```
755 \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.

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

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

```
757 \newcommand*\glsSetSuffixFF[1]{%
```

```
758 \renewcommand*\gls@suffixFF{#1}%
```

```
759 }
```

`\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 `\glshypernumber`, otherwise it will simply display its argument "as is".

```
760 \ifcsundef{hyperlink}%
```

```
761 {%
```

```
762 \newcommand*\glsnumberformat[1]{#1}%
```

```
763 }%
```

```
764 {%
```

```
765 \newcommand*\glsnumberformat[1]{\glshypernumber{#1}}%
```

```
766 }
```

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`

```
767 \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`

```
768 \newcommand{\delimR}{--}
```

The glossary preamble is given by `\glossarypreamble`. This will appear after the glossary sectioning command, and before the `\theglossary` 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`

```
769 \newcommand*{\glossarypreamble}{%
770   \csuse{@glossarypreamble@\currentglossary}%
771 }
```

`\setglossarypreamble`

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

Code provided by Michael Pock.

```
772 \newcommand{\setglossarypreamble}[2][\glsdefaulttype]{%
773   \ifglossaryexists{#1}{%
774     \csgdef{@glossarypreamble@#1}{#2}%
775   }{%
776     \GlossariesWarning{%
777       Glossary ‘#1’ is not defined%
778     }%
779   }%
780 }
```

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`

```
781 \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`.


```

782 \newcommand*{\glossarysection}[2][\@gls@title]{%
783   \def\@gls@title{#2}%
784   \ifcsundef{phantomsection}%
785   {%
786     \@glossarysection{#1}{#2}%
787   }%
788   {%
789     \@pglossarysection{#1}{#2}%
790   }%

791   \glsglossarymark{\glossarytoctitle}%
792 }

```

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

```

793 \ifcsundef{glossarymark}%
794 {%
795   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}
796 }%
797 {%
798   \@ifclassloaded{memoir}
799   {%
800     \newcommand{\glsglossarymark}[1]{%
801       \ifglsucmark
802         \markboth{\memUHead{#1}}{\memUHead{#1}}%
803       \else
804         \markboth{#1}{#1}%
805       \fi
806     }
807   }%
808   {%
809     \newcommand{\glsglossarymark}[1]{%
810       \ifglsucmark
811         \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
812       \else
813         \@mkboth{#1}{#1}%
814       \fi
815     }
816   }
817 }

```

`\glossarymark` Provided for backward compatibility:

```

818 \providecommand{\glossarymark}[1]{%
819   \ifglsucmark
820     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
821   \else
822     \@mkboth{#1}{#1}%
823   \fi
824 }

```

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`

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

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

`\@glossarysection`

```
827 \newcommand*\@glossarysection}[2]{%
828   \ifdefempty\@@glossarysecstar
829   {%
830     \csname\@@glossarysec\endcsname{#2}%
831   }%
832   {%
833     \csname\@@glossarysec\endcsname*{#2}%
834     \@gls@toc{#1}{\@@glossarysec}%
835   }%
```

Do automatic labelling if required

```
836   \@@glossaryseclabel
837 }
```

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.

`\@pglossarysection`

```
838 \newcommand*\@pglossarysection}[2]{%
839   \glsclearpage
840   \phantomsection
841   \ifdefempty\@@glossarysecstar
842   {%
843     \csname\@@glossarysec\endcsname{#2}%
844   }%
845   {%
846     \@gls@toc{#1}{\@@glossarysec}%
847     \csname\@@glossarysec\endcsname*{#2}%
848   }%
```

Do automatic labelling if required

```
849   \@@glossaryseclabel
850 }
```

`\gls@doclearpage` The `\gls@doclearpage` 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.

```

851 \newcommand*{\gls@doclearpage}{%
852   \ifthenelse{\equal{\@glossarysec}{chapter}}{%
853     {%
854       \ifcsundef{cleardoublepage}%
855       {%
856         \clearpage
857       }%
858     {%
859       \ifcsdef{if@openright}%
860       {%
861         \if@openright
862           \cleardoublepage
863         \else
864           \clearpage
865         \fi
866       }%
867     {%
868       \cleardoublepage
869     }%
870   }%
871 }%
872 {}%
873 }

```

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

```

874 \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`

```

875 \newcommand*{\@gls@toc}[2]{%
876   \ifglstoc
877     \ifglslnumberline
878       \addcontentsline{toc}{#2}{\numberline{#1}}%
879     \else
880       \addcontentsline{toc}{#2}{#1}%
881     \fi
882   \fi
883 }

```

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

```
884 \newcommand*{\glsnoxindywarning}[1]{%
885   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
886 }
```

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

```
887 \ifglsxindy
888   \edef\@xdyattributes{\string"default\string"}%
889 \fi
```

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

```
890 \ifglsxindy
891   \edef\@xdyattributelist{}%
892 \fi
```

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

```
893 \ifglsxindy
894   \def\@xdylocref{}
895 \fi
```

`\@gls@ifinlist`

```
896 \newcommand*{\@gls@ifinlist}[4]{%
897   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
898     \def\@gls@listsuffix{##2}%
899     \ifx\@gls@listsuffix\@empty
900       #4%
901     \else
902       #3%
903     \fi
904   }%
905   \@do@ifinlist,#2,#1,\end@do@ifinlist
906 }
```

`\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.

```
907 \ifglsxindy
908   \newcommand*{\@xdycounters}{\glscounter}
909   \newcommand*\GlsAddXdyCounters[1]{%
910     \@for\@gls@ctr:=#1\do{%
```

Check if already in list before adding.

```

911     \edef\@do@addcounter{%
912         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
913         {%
914             \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
915                 \noexpand\@gls@ctr}%
916         }%
917     }%
918     \@do@addcounter
919 }
920 }

```

Only has an effect before `\writeist`:

```

921 \@onlypremakeg\GlsAddXdyCounters
922 \else
923 \newcommand*\GlsAddXdyCounters[1]{%
924     \glsnoxindywarning\GlsAddXdyAttribute
925 }
926 \fi

```

`\d@glssaddxdycounters` Counters must all be identified before adding attributes.

```

927 \newcommand*\@disabled@glssaddxdycounters{%
928     \PackageError{glossaries}{\string\GlsAddXdyCounters\space
929     can't be used after \string\GlsAddXdyAttribute}{Move all
930     occurrences of \string\GlsAddXdyCounters\space before the first
931     instance of \string\GlsAddXdyAttribute}%
932 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

933 \ifglsxindy

```

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

```

934 \newcommand*\@glssaddxdyattribute[2]{%

```

Add to xindy attribute list

```

935     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
936         \string"#2#1\string"}%

```

Add to xindy markup location.

```

937     \expandafter\toks@\expandafter{\@xdylocref}%
938     \edef\@xdylocref{\the\toks@ ^^J%
939         (markup-locref
940         :open \string"\string~n%
941         \expandafter\string\csname glsX#2X#1\endcsname
942         \string" ^^J
943         :close \string"\string" ^^J
944         :attr \string"#2#1\string")}%

```

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

```

945     \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%

```

```

946     \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
947   }%
948 }

```

High-level command:

```

949 \newcommand*\GlsAddXdyAttribute[1]{%

```

Add to comma-separated attribute list

```

950   \ifx\@xdyattributelist\@empty
951     \edef\@xdyattributelist{#1}%
952   \else
953     \edef\@xdyattributelist{\@xdyattributelist,#1}%
954   \fi

```

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

```

955   \@for\@this@counter:=\@xdycounters\do{%
956     \protected@edef\gls@do@addxdyattribute{%
957       \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
958     }
959     \gls@do@addxdyattribute
960   }%

```

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

```

961   \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
962 }

```

Only has an effect before `\writeist`:

```

963 \@onlypremakeg\GlsAddXdyAttribute
964 \else
965   \newcommand*\GlsAddXdyAttribute[1]{%
966     \glsnoxindywarning\GlsAddXdyAttribute}
967 \fi

```

redefinedattributes Add known attributes for all defined counters

```

968 \ifglxsindy
969 \newcommand*{\@gls@addpredefinedattributes}{%
970   \GlsAddXdyAttribute{glsnumberformat}
971   \GlsAddXdyAttribute{textrm}
972   \GlsAddXdyAttribute{textsf}
973   \GlsAddXdyAttribute{texttt}
974   \GlsAddXdyAttribute{textbf}
975   \GlsAddXdyAttribute{textmd}
976   \GlsAddXdyAttribute{textit}
977   \GlsAddXdyAttribute{textup}
978   \GlsAddXdyAttribute{textsl}
979   \GlsAddXdyAttribute{textsc}
980   \GlsAddXdyAttribute{emph}
981   \GlsAddXdyAttribute{glshypernumber}
982   \GlsAddXdyAttribute{hyperrrm}
983   \GlsAddXdyAttribute{hypersf}
984   \GlsAddXdyAttribute{hypertt}

```

```

985 \GlsAddXdyAttribute{hyperbf}
986 \GlsAddXdyAttribute{hypermd}
987 \GlsAddXdyAttribute{hyperit}
988 \GlsAddXdyAttribute{hyperup}
989 \GlsAddXdyAttribute{hypersl}
990 \GlsAddXdyAttribute{hypersc}
991 \GlsAddXdyAttribute{hyperemph}
992 }
993 \else
994 \let\@gls@addpredefinedattributes\relax
995 \fi

```

`\@xdyuseralphabets` List of additional alphabets

```

996 \def\@xdyuseralphabets{}

```

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

```

997 \ifglsxindy
998 \newcommand*{\GlsAddXdyAlphabet}[2]{%
999 \edef\@xdyuseralphabets{%
1000 \@xdyuseralphabets ^^J
1001 (define-alphabet "#1" (#2))}}
1002 \else
1003 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1004 \glsnoxindywarning\GlsAddXdyAlphabet}
1005 \fi

```

This code is only required for xindy:

```

1006 \ifglsxindy

```

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

```

1007 \newcommand*{\@gls@xdy@locationlist}{%
1008 roman-page-numbers,%
1009 Roman-page-numbers,%
1010 arabic-page-numbers,%
1011 alpha-page-numbers,%
1012 Alpha-page-numbers,%
1013 Appendix-page-numbers,%
1014 arabic-section-numbers%
1015 }

```

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.

```

1016 \protected\edef\@gls@roman{\@roman{0}\string"

```

```

1017     \string"roman-numbers-lowercase\string" :sep \string"}}%
1018 \@onelevel@sanitize\@gls@roman
1019 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
1020     :sep \string"}}%
1021 \@onelevel@sanitize\@tmp
1022 \ifx\@tmp\@gls@roman
1023     \expandafter
1024     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
1025         \string"roman-numbers-lowercase\string"%
1026     }%
1027 \else
1028     \expandafter
1029     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
1030         :sep \string"\@gls@roman\string"%
1031     }%
1032 \fi

```

@Roman-page-numbers Upper case Roman numerals (I, II, ...).

```

1033 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
1034     \string"roman-numbers-uppercase\string"%
1035 }%

```

arabic-page-numbers Arabic numbers (1, 2, ...).

```

1036 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
1037     \string"arabic-numbers\string"%
1038 }%

```

@alpha-page-numbers Lower case alphabetical (a, b, ...).

```

1039 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
1040     \string"alpha\string"%
1041 }%

```

@Alpha-page-numbers Upper case alphabetical (A, B, ...).

```

1042 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
1043     \string"ALPHA\string"%
1044 }%

```

pendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given by \@glsAlphacompositor.

```

1045 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1046     \string"ALPHA\string"
1047     :sep \string"\@glsAlphacompositor\string"
1048     \string"arabic-numbers\string"%
1049 }

```

bic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by \glscompositor.

```

1050 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%

```



```

1051 \string"arabic-numbers\string"
1052 :sep \string"\glscompositor\string"
1053 \string"arabic-numbers\string"%
1054 }%

```

`\xdyuserlocationdefs` List of additional location definitions (separated by `^^J`)

```

1055 \def\xdyuserlocationdefs{}

```

`\xdyuserlocationnames` List of additional user location names

```

1056 \def\xdyuserlocationnames{}

```

End of xindy-only block:

```

1057 \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.)

```

1058 \ifglxsxindy
1059 \newcommand*\GlsAddXdyLocation[3][{}]{%
1060 \def\@gls@tmp{#1}%
1061 \ifx\@gls@tmp\@empty
1062 \edef\xdyuserlocationdefs{%
1063 \xdyuserlocationdefs ^^J%
1064 (define-location-class \string"#2\string"^^J\space\space
1065 \space(:sep \string"{}\glsopenbrace\string" #3
1066 :sep \string"\glsclosebrace\string"))
1067 }%
1068 \else
1069 \edef\xdyuserlocationdefs{%
1070 \xdyuserlocationdefs ^^J%
1071 (define-location-class \string"#2\string"^^J\space\space
1072 \space(:sep "\glsopenbrace"
1073 #1
1074 :sep "\glsclosebrace\glsopenbrace" #3
1075 :sep "\glsclosebrace"))
1076 }%
1077 \fi
1078 \edef\xdyuserlocationnames{%
1079 \xdyuserlocationnames^^J\space\space\space
1080 \string"#1\string"}%
1081 }

```

Only has an effect before `\writeist`:

```

1082 \@onlypremake\GlsAddXdyLocation
1083 \else
1084 \newcommand*\GlsAddXdyLocation[2]{%
1085 \glsnnoxindywarning\GlsAddXdyLocation}
1086 \fi

```

ylocationclassorder Define location class order

```
1087 \ifglxindy
1088   \edef\@xdylocationclassorder{^^J\space\space\space
1089     \string"roman-page-numbers\string"^^J\space\space\space
1090     \string"arabic-page-numbers\string"^^J\space\space\space
1091     \string"arabic-section-numbers\string"^^J\space\space\space
1092     \string"alpha-page-numbers\string"^^J\space\space\space
1093     \string"Roman-page-numbers\string"^^J\space\space\space
1094     \string"Alpha-page-numbers\string"^^J\space\space\space
1095     \string"Appendix-page-numbers\string"
1096     \@xdyuserlocationnames^^J\space\space\space
1097     \string"see\string"
1098   }
1099 \fi
```

Change the location order.

yLocationClassOrder

```
1100 \ifglxindy
1101   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1102     \def\@xdylocationclassorder{#1}}
1103 \else
1104   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1105     \glsnxindywarning\GlsSetXdyLocationClassOrder}
1106 \fi
```

\@xdysortrules Define sort rules

```
1107 \ifglxindy
1108   \def\@xdysortrules{}
1109 \fi
```

\GlsAddSortRule Add a sort rule

```
1110 \ifglxindy
1111   \newcommand*\GlsAddSortRule[2]{%
1112     \expandafter\toks@\expandafter{\@xdysortrules}%
1113     \protected@edef\@xdysortrules{\the\toks@ ^^J
1114       (sort-rule \string"#1\string" \string"#2\string")}%
1115   }
1116 \else
1117   \newcommand*\GlsAddSortRule[2]{%
1118     \glsnxindywarning\GlsAddSortRule}
1119 \fi
```

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

```
1120 \ifglxindy
1121   \def\@xdyrequiredstyles{tex}
1122 \fi
```

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

```
1123 \ifglxindy
1124   \newcommand*\GlsAddXdyStyle[1]{%
1125     \edef\xdyrequiredstyles{\xdyrequiredstyles,#1}}%
1126 \else
1127   \newcommand*\GlsAddXdyStyle[1]{%
1128     \glsnnoxindywarning\GlsAddXdyStyle}
1129 \fi
```

`\GlsSetXdyStyles` Reset the list of required styles

```
1130 \ifglxindy
1131   \newcommand*\GlsSetXdyStyles[1]{%
1132     \edef\xdyrequiredstyles{#1}}
1133 \else
1134   \newcommand*\GlsSetXdyStyles[1]{%
1135     \glsnnoxindywarning\GlsSetXdyStyles}
1136 \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.

```
1137 \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.

```
1138 \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.

```
1139 \ifglxindy
1140   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1141     \ifglossaryexists{#1}{%
1142       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1143     }{%
1144       \PackageError{glossaries}{Can't set language type for
1145         glossary type '#1' --- no such glossary}{%
1146         You have specified a glossary type that doesn't exist}}
1147 \else
1148   \newcommand*\GlsSetXdyLanguage[2][]{%
1149     \glsnnoxindywarning\GlsSetXdyLanguage}
1150 \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.

```
1151 \def\@gls@codepage#1#2{}
```

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

```
1152 \ifglsxindy
1153   \newcommand*\GlsSetXdyCodePage[1]{%
1154     \renewcommand*\@gls@codepage{#1}%
1155   }
```

Suggested by egreg:

```
1156   \AtBeginDocument{%
1157     \ifx\@gls@codepage\@empty
1158       \ifpackageloaded{fontspec}{\def\@gls@codepage{utf8}}{}%
1159     \fi
1160   }
1161 \else
1162   \newcommand*\GlsSetXdyCodePage[1]{%
1163     \glsnoxywarning\GlsSetXdyCodePage%
1164   \fi
```

`\@xdylettergroups` Store letter group definitions.

```
1165 \ifglsxindy
1166   \ifglsxindy@glslnumbers
1167     \def\@xdylettergroups{(define-letter-group
1168       \string"glslnumbers\string"^^J\space\space\space
1169       :prefixes (\string"0\string" \string"1\string"
1170       \string"2\string" \string"3\string" \string"4\string"
1171       \string"5\string" \string"6\string" \string"7\string"
1172       \string"8\string" \string"9\string")^^J\space\space\space
1173       :before \string"\@glsfirstletter\string")}
1174   \else
1175     \def\@xdylettergroups{}
1176   \fi
1177 \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.

```
1178   \newcommand*\GlsAddLetterGroup[2]{%
1179     \expandafter\toks@\expandafter{\@xdylettergroups}%
1180     \protected@edef\@xdylettergroups{\the\toks@^^J%
1181     (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
1182   }
```

1.5 Loops and conditionals

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

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

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

```
1183 \newcommand*\forallglossaries}[3][\@glo@types]{%
1184   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1185 }
```

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

```
\forall glossentries[<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.

```
1186 \newcommand*\forallglossentries}[3][\glsdefaulttype]{%
1187   \edef\@glo@list{\csname glolist@#1\endcsname}%
1188   \@for#2:=\@glo@list\do
1189     {%
1190       \ifdefempty{#2}{-}{#3}%
1191     }%
1192 }
```

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

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

Within `\forall glossentries`, the current glossary type is given by `\@this@glo@`.

```
1193 \newcommand*\forallglossentries}[3][\@glo@types]{%
1194   \expandafter\forallglossaries\expandafter[#1]{\@this@glo@}%
1195   {%
1196     \forallglossentries[\@this@glo@]{#2}{#3}%
1197   }%
1198 }
```

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

```
\if glossaryexists{<type>}{<true-text>}{<false-text>}
```

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

```
1199 \newcommand{\ifglossaryexists}[3]{%
1200   \ifcsundef{@glo@type@#1@out}{#3}{#2}%
1201 }
```

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`

```
1202 \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.

```
1203 \newcommand{\ifglsentryexists}[3]{%
1204   \ifcsundef{glo@\glsdetoklabel{#1}@name}{#3}{#2}%
1205 }
```

`\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>`.

```
1206 \newcommand*\ifglsused}[3]{%
1207   \ifbool{glo@\glsdetoklabel{#1}@flag}{#2}{#3}%
1208 }
```

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>`.

```
1209 \newcommand{\glsdoifexists}[2]{%
1210   \ifglsentryexists{#1}{#2}{%

```

```

1211 \PackageError{glossaries}{Glossary entry ‘\glsdetoklabel{#1}’
1212 has not been defined}{You need to define a glossary entry before you
1213 can use it.}}%
1214 }

```

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

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

```

1215 \newcommand{\glsdoifnoexists}[2]{%
1216 \ifglstryexists{#1}{%
1217 \PackageError{glossaries}{Glossary entry ‘\glsdetoklabel{#1}’ has already
1218 been defined}{}}{#2}%
1219 }

```

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

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

```

1220 \newcommand{\glsdoifexistsorwarn}[2]{%
1221 \ifglstryexists{#1}{#2}{%
1222 \GlossariesWarning{Glossary entry ‘\glsdetoklabel{#1}’
1223 has not been defined}%
1224 }%
1225 }

```

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

```

1226 \newcommand{\ifglshaschildren}[3]{%
1227 \glsdoifexists{#1}%
1228 {%
1229 \def\do@glshaschildren{#3}%
1230 \edef\@gls@thislabel{\glsdetoklabel{#1}}%
1231 \expandafter\forlstryentries\expandafter
1232 [\csname glo@\@gls@thislabel @type\endcsname]
1233 {\glo@label}%
1234 {%
1235 \letcs\glo@parent{glo@\glo@label @parent}%
1236 \ifdefequal\@gls@thislabel\glo@parent
1237 {%
1238 \def\do@glshaschildren{#2}%
1239 \@endfortrue
1240 }%
1241 }%
1242 }%
1243 \do@glshaschildren
1244 }%
1245 }

```

```

\ifglshasparent \ifglshaschildren{<label>}{<true part>}{<false part>}
1246 \newcommand{\ifglshasparent}[3]{%
1247   \glsdoifexists{#1}%
1248   {%
1249     \ifcseempty{glo@\glsdetoklabel{#1}@parent}{#3}{#2}%
1250   }%
1251 }

\ifglshasdesc \ifglshasdesc{<label>}{<true part>}{<false part>}
1252 \newcommand*{\ifglshasdesc}[3]{%
1253   \ifcseempty{glo@\glsdetoklabel{#1}@desc}%
1254   {#3}%
1255   {#2}%
1256 }

\ifglsdessuppressed \ifglsdessuppressed{<label>}{<true part>}{<false part>} Does <true part>
if the description is just \nopostdesc otherwise does <false part>.
1257 \newcommand*{\ifglsdessuppressed}[3]{%
1258   \ifcsequal{glo@\glsdetoklabel{#1}@desc}{@no@post@desc}%
1259   {#2}%
1260   {#3}%
1261 }

\ifglshassymbol \ifglshassymbol{<label>}{<true part>}{<false part>}
1262 \newcommand*{\ifglshassymbol}[3]{%
1263   \letcs{\@glo@symbol}{glo@\glsdetoklabel{#1}@symbol}%
1264   \ifdefempty\@glo@symbol
1265   {#3}%
1266   {%
1267     \ifdefequal\@glo@symbol\@gls@default@value
1268     {#3}%
1269     {#2}%
1270   }%
1271 }

\ifglshaslong \ifglshaslong{<label>}{<true part>}{<false part>}
1272 \newcommand*{\ifglshaslong}[3]{%
1273   \letcs{\@glo@long}{glo@\glsdetoklabel{#1}@long}%
1274   \ifdefempty\@glo@long
1275   {#3}%
1276   {%
1277     \ifdefequal\@glo@long\@gls@default@value
1278     {#3}%
1279     {#2}%
1280   }%
1281 }

\ifglshasshort \ifglshasshort{<label>}{<true part>}{<false part>}

```



```

1282 \newcommand*{\ifglshasshort}[3]{%
1283   \letcs{\@glo@short}{glo@glstdetoklabel{#1}@short}%
1284   \ifdefempty\@glo@short
1285     {#3}%
1286     {%
1287       \ifdefequal\@glo@short\@gls@default@value
1288         {#3}%
1289         {#2}%
1290     }%
1291 }

```

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

```

1292 \newcommand*{\ifglshasfield}[4]{%
1293   \glsdoifexists{#2}%
1294   {%
1295     \letcs{\@glo@thisvalue}{glo@glstdetoklabel{#2}@#1}%

```

First check supplied field label is defined.

```

1296     \ifdef\@glo@thisvalue
1297     {%

```

Is defined, so now check if empty.

```

1298         \ifdefempty\@glo@thisvalue
1299         {%

```

Is empty, so doesn't have field set.

```

1300             #4%
1301         }%
1302     {%

```

Not empty, so check if set to \@gls@default@value

```

1303         \ifdefequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1304     }%
1305 }%
1306 {%

```

Field given isn't defined, so check if mapping exists.

```

1307     \@gls@fetchfield{\@gls@thisfield}{#1}%

```

If \@gls@thisfield is defined, we've found a map. If not, the field supplied doesn't exist.

```

1308     \ifdef\@gls@thisfield
1309     {%

```

Is defined, so now check if empty.

```

1310         \letcs{\@glo@thisvalue}{glo@glstdetoklabel{#2}@ \@gls@thisfield}%
1311         \ifdefempty\@glo@thisvalue
1312         {%

```

Is empty so field hasn't been set.

```
1313         #4%
1314     }%
1315     {%
```

Isn't empty so check if it's been set to \@gls@default@value.

```
1316         \ifdefequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1317     }%
1318     }%
1319     {%
```

Not defined.

```
1320         \GlossariesWarning{Unknown entry field '#1'}%
1321         #4%
1322     }%
1323 }%
1324 }%
1325 }
```

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

```
1326 \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.

```
1327 \newcommand*{\@gls@provide@newglossary{%
1328     \protected@write\@auxout{}{\string\providecommand\string\@newglossary[4]{}}%
```

Only need to do this once.

```
1329     \let\@gls@provide@newglossary\relax
1330 }
```

\defglsentryfmt Allow different glossaries to have different display styles.

```
1331 \newcommand*{\defglsentryfmt}[2][\glsdefaulttype]{%
1332     \csgdef{gls@#1@entryfmt}{#2}%
1333 }
```

\gls@doentryfmt

```
1334 \newcommand*{\gls@doentryfmt}[1]{\csuse{gls@#1@entryfmt}}
```

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`

```
1335 \newcommand*{\newglossary}[5][glg]{%
1336   \ifglossaryexists{#2}%
1337   {%
1338     \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
1339       You can’t define a new glossary called ‘#2’ because it already
1340       exists}%
1341   }%
1342   {%
    Check if default has been set
1343     \ifundef\glsdefaulttype
1344     {%
1345       \gdef\glsdefaulttype{#2}%
1346     }{}%
    Add this to the list of glossary types:
1347     \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.
1348     \expandafter\gdef\csname glolist@#2\endcsname{,}%
    Store details of this new glossary type:
1349     \expandafter\def\csname @glotype@#2in\endcsname{#3}%
1350     \expandafter\def\csname @glotype@#2out\endcsname{#4}%
1351     \expandafter\def\csname @glotype@#2title\endcsname{#5}%
1352     \@gls@provide@newglossary
1353     \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.

```

1354 \ifcsundef{gls@#2@entryfmt}%
1355 {%
1356     \defglsentryfmt[#2]{\glsentryfmt}%
1357 }%
1358 {}%

```

Define sort counter if required:

```

1359 \@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.)

```

1360 \@ifnextchar[{\@gls@setcounter{#2}}%
1361     {\@gls@setcounter{#2}[\glscounter]}}%
1362 }

```

`\altnewglossary`

```

1363 \newcommand*{\altnewglossary}[3]{%
1364     \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1365 }

```

Only define new glossaries in the preamble:

```

1366 \@onlypreamble{\newglossary}

```

Only define new glossaries before `\makeglossaries`

```

1367 \@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 \TeX , `\@newglossary` simply ignores its arguments.

`\@newglossary`

```

1368 \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`

```

1369 \def\@gls@setcounter#1[#2]{%
1370     \expandafter\def\csname @glotype@#1@counter\endcsname{#2}%

```

Add counter to xindy list, if not already added:

```

1371     \ifglxindy
1372         \GlsAddXdyCounters{#2}%
1373     \fi
1374 }

```

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

`\@gls@getcounter`

```

1375 \newcommand*{\@gls@getcounter}[1]{%
1376     \csname @glotype@#1@counter\endcsname
1377 }

```

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

```
1378 \glsdefmain
```

Define the “acronym” glossaries if required.

```
1379 \@gls@do@acronymsdef
```

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

```
1380 \@gls@do@symbolsdef
```

```
1381 \@gls@do@numbersdef
```

```
1382 \@gls@do@indexdef
```

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.

```
1383 \define@key{glossentry}{name}{%
```

```
1384 \def\@glo@name{#1}}%
```

```
1385 }
```

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

```
1386 \define@key{glossentry}{description}{%
```

```
1387 \def\@glo@desc{#1}}%
```

```
1388 }
```

descriptionplural

```
1389 \define@key{glossentry}{descriptionplural}{%
```

```
1390 \def\@glo@descplural{#1}}%
```

```
1391 }
```

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`.

```
1392 \define@key{glossentry}{sort}{%
```

```
1393 \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.

```
1394 \define@key{glossentry}{text}{%
1395 \def\@glo@text{#1}%
1396 }
```

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.

```
1397 \define@key{glossentry}{plural}{%
1398 \def\@glo@plural{#1}%
1399 }
```

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.

```
1400 \define@key{glossentry}{first}{%
1401 \def\@glo@first{#1}%
1402 }
```

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.

```
1403 \define@key{glossentry}{firstplural}{%
1404 \def\@glo@firstplural{#1}%
1405 }
```

`\@gls@default@value`

```
1406 \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).

```
1407 \define@key{glossentry}{symbol}{%
1408 \def\@glo@symbol{#1}%
1409 }
```

symbolplural

```
1410 \define@key{glossentry}{symbolplural}{%
1411 \def\@glo@symbolplural{#1}%
1412 }
```

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

```
1413 \define@key{glossentry}{type}{%
1414 \def\@glo@type{#1}}
```

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

```
1415 \define@key{glossentry}{counter}{%
1416 \ifcsundef{c@#1}%
1417 {%
1418 \PackageError{glossaries}%
1419 {There is no counter called ‘#1’}%
1420 {%
1421 The counter key should have the name of a valid counter
1422 as its value%
1423 }%
1424 }%
1425 {%
1426 \def\@glo@counter{#1}%
1427 }%
1428 }
```

see The see key specifies a list of cross-references

```
1429 \define@key{glossentry}{see}{%
1430 \gls@checkseeallowed
1431 \def\@glo@see{#1}%
1432 \@glo@seeautonumberlist
1433 }
```

gls@checkseeallowed

```
1434 \newcommand*{\gls@checkseeallowed}{%
1435 \PackageError{glossaries}%
1436 {‘see’ key may only be used after \string\makeglossaries}%
1437 {You must use \string\makeglossaries\space before defining
1438 any entries that have a ‘see’ key}%
1439 }
```

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

```
1440 \define@key{glossentry}{parent}{%
1441 \def\@glo@parent{#1}}
```

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

```
1442 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1443 \ifcase\nr\relax
1444 \def\@glo@prefix{\glsnonextpages}%
1445 \else
1446 \def\@glo@prefix{\glsnextpages}%
1447 }
```

```

1447 \fi
1448 }

```

Define some generic user keys. (6 ought to be enough!)

user1

```

1449 \define@key{glossentry}{user1}{%
1450 \def\@glo@useri{#1}%
1451 }

```

user2

```

1452 \define@key{glossentry}{user2}{%
1453 \def\@glo@userii{#1}%
1454 }

```

user3

```

1455 \define@key{glossentry}{user3}{%
1456 \def\@glo@useriii{#1}%
1457 }

```

user4

```

1458 \define@key{glossentry}{user4}{%
1459 \def\@glo@useriv{#1}%
1460 }

```

user5

```

1461 \define@key{glossentry}{user5}{%
1462 \def\@glo@userv{#1}%
1463 }

```

user6

```

1464 \define@key{glossentry}{user6}{%
1465 \def\@glo@uservi{#1}%
1466 }

```

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.

```

1467 \define@key{glossentry}{short}{%
1468 \def\@glo@short{#1}%
1469 }

```

shortplural This key is provided for use by \newacronym.

```

1470 \define@key{glossentry}{shortplural}{%
1471 \def\@glo@shortpl{#1}%
1472 }

```

long This key is provided for use by \newacronym.

```

1473 \define@key{glossentry}{long}{%
1474 \def\@glo@long{#1}%
1475 }

```


longplural This key is provided for use by \newacronym.

```
1476 \define@key{glossentry}{longplural}{%
1477   \def\@glo@longpl{#1}%
1478 }
```

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

```
1479 \newcommand*{\@glsnname}{%
1480   \PackageError{glossaries}{name key required in
1481     \string\newglossaryentry\space for entry '\@glo@label'}{You
1482     haven't specified the entry name}}
```

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

```
1483 \newcommand*{\@glsnodelsc}{%
1484   \PackageError{glossaries}
1485   {%
1486     description key required in \string\newglossaryentry\space
1487     for entry '\@glo@label'%
1488   }%
1489   {%
1490     You haven't specified the entry description%
1491   }%
1492 }%
```

\@glsdefaultplural Now obsolete. Don't use.

```
1493 \newcommand*{\@glsdefaultplural}{{}}
```

s@missingnumberlist Define a command to generate warning when numberlist not set.

```
1494 \newcommand*{\@gls@missingnumberlist}[1]{%
1495   ??%
1496   \ifglssavenumberlist
1497     \GlossariesWarning{Missing number list for entry '#1'.
1498       Maybe makeglossaries + rerun required.}%
1499   \else
1500     \PackageError{glossaries}%
1501     {Package option 'savenumberlist=true' required.}%
1502     {%
1503       You must use the 'savenumberlist' package option
1504       to reference location lists.%
1505     }%
1506   \fi
1507 }
```

\@glsdefaultsort Define command to set default sort.

```
1508 \newcommand*{\@glsdefaultsort}{\@glo@name}
```

\gls@level Register to increment entry levels.

```
1509 \newcount\gls@level
```

@gls@noexpand@field

```
1510 \newcommand{\@gls@noexpand@field}[3]{%
1511 \expandafter\global\expandafter
1512 \let\csname glo@#1@#2\endcsname#3%
1513 }
```

gls@noexpand@fields

```
1514 \newcommand{\@gls@noexpand@fields}[4]{%
1515 \ifcsdef{gls@assign@#3@field}
1516 {%
1517 \ifdefequal{#4}{\@gls@default@value}%
1518 {%
1519 \edef\@gls@value{\expandonce{#1}}%
1520 \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1521 }%
1522 {%
1523 \csuse{gls@assign@#3@field}{#2}{#4}%
1524 }%
1525 }%
1526 {%
1527 \ifdefequal{#4}{\@gls@default@value}%
1528 {%
1529 \edef\@gls@value{\expandonce{#1}}%
1530 \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1531 }%
1532 {%
1533 \@gls@noexpand@field{#2}{#3}{#4}%
1534 }%
1535 }%
1536 }
```

\@gls@expand@field

```
1537 \newcommand{\@gls@expand@field}[3]{%
1538 \expandafter
1539 \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1540 }
```

@gls@expand@fields

```
1541 \newcommand{\@gls@expand@fields}[4]{%
1542 \ifcsdef{gls@assign@#3@field}
1543 {%
1544 \ifdefequal{#4}{\@gls@default@value}%
1545 {%
1546 \edef\@gls@value{\expandonce{#1}}%
1547 \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1548 }%
1549 {%
1550 \expandafter\@gls@startswitexpandonce#4\relax\relax\gls@endcheck
```

```

1551      {%
1552      \@@gls@expand@field{#2}{#3}{#4}%
1553      }%
1554      {%
1555      \csuse{gls@assign@#3@field}{#2}{#4}%
1556      }%
1557      }%
1558      }%
1559      {%
1560      \ifdefequal{#4}{\@gls@default@value}%
1561      {%
1562      \@@gls@expand@field{#2}{#3}{#1}%
1563      }%
1564      {%
1565      \@@gls@expand@field{#2}{#3}{#4}%
1566      }%
1567      }%
1568      }

```

startswithexpandonce

```

1569 \def\@gls@expandonce{\expandonce}
1570 \def\@gls@startswithexpandonce#1#2\gls@endcheck#3#4{%
1571   \def\@gls@tmp{#1}%
1572   \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%
1573 }

```

`\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.

```
1574 \let\gls@assign@field\@gls@expand@fields
```

`\glsexpandfields` Fully expand values when assigning fields (except for specific fields that are overridden by `\glssetnoexpandfield`).

```

1575 \newcommand*\glsexpandfields{%
1576   \let\gls@assign@field\@gls@expand@fields
1577 }

```

`\glsnoexpandfields` Don't expand values when assigning fields (except for specific fields that are overridden by `\glssetexpandfield`).

```

1578 \newcommand*\glsnoexpandfields{%
1579   \let\gls@assign@field\@gls@noexpand@fields
1580 }

```

`\newglossaryentry` Define `\newglossaryentry {<label>}{<key-val list>}`. There are two required fields in `<key-val list>`: name (or parent) and description. (See above.)

```
1581 \newrobustcmd{\newglossaryentry}[2]{%
```

Check to see if this glossary entry has already been defined:

```
1582 \glsdoifnoexists{#1}%
1583 {%
1584     \gls@defglossaryentry{#1}{#2}%
1585 }%
1586 }
```

`\provideglossaryentry` Like `\newglossaryentry` but does nothing if the entry has already been defined.

```
1587 \newrobustcmd{\provideglossaryentry}[2]{%
1588     \ifglsentryexists{#1}%
1589     }{%
1590     {%
1591         \gls@defglossaryentry{#1}{#2}%
1592     }%
1593 }
1594 \@onlypreamble{\provideglossaryentry}
```

`\new@glossaryentry` For use in document environment.

```
1595 \newrobustcmd{\new@glossaryentry}[2]{%
1596     \ifundef\@gls@deffile
1597     {%
1598         \global\newwrite\@gls@deffile
1599         \immediate\openout\@gls@deffile=\jobname.glsdefs
1600     }%
1601     }{%
1602     \ifglsentryexists{#1}{}%
1603     {%
1604         \gls@defglossaryentry{#1}{#2}%
1605     }%
1606     \@gls@writedef{#1}%
1607 }
1608 \AtBeginDocument
1609 {
1610     \makeatletter
1611     \InputIfFileExists{\jobname.glsdefs}{\}{}%
1612     \makeatother
1613     \let\newglossaryentry\new@glossaryentry
1614 }
1615 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}{}}
1616 % \end{macrocode}
1617 %\end{macro}
1618 %
1619 %\begin{macro}{\@gls@writedef}
1620 %\changes{3.10a}{2013-10-13}{new}
1621 % Writes glossary entry definition to \cs{@gls@deffile}.
1622 %\changes{4.03}{2014-01-20}{added \cs{glsdetoklabel}}
```

```

1623% \begin{macrocode}
1624 \newcommand*{\@gls@writedef}[1]{%
1625 \immediate\write\@gls@deffile
1626 {%
1627 \string\ifglsentryexists{#1}{}\expandafter\@gobble\string\%^~J%
1628 \expandafter\@gobble\string\{\expandafter\@gobble\string\%^~J%
1629 \string\gls@defglossaryentry{\glsdetoklabel{#1}}\expandafter
1630 \@gobble\string\%^~J%
1631 \expandafter\@gobble\string\{\expandafter\@gobble\string\%%
1632 }%

Write key value information:

1633 \@for\@gls@map:=\@gls@keymap\do
1634 {%
1635 \edef\glo@value{\expandafter\expandonce
1636 \csname glo@\glsdetoklabel{#1}\expandafter
1637 \@secondoftwo\@gls@map\endcsname}%
1638 \@onelevel@sanitize\glo@value
1639 \immediate\write\@gls@deffile
1640 {%
1641 \expandafter\@firstoftwo\@gls@map
1642 =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1643 \expandafter\@gobble\string\%%
1644 }%
1645 }%

Provide hook:

1646 \gls@writedefhook
1647 \immediate\write\@gls@deffile
1648 {%
1649 \expandafter\@gobble\string\%^~J%
1650 \expandafter\@gobble\string\}\expandafter\@gobble\string\%^~J%
1651 \expandafter\@gobble\string\}\expandafter\@gobble\string\%%
1652 }%
1653 }

```

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

```

1654 \newcommand*{\@gls@keymap}{%
1655 {name}{name},%
1656 {sort}{sortvalue},% unescaped sort value
1657 {type}{type},%
1658 {first}{first},%
1659 {firstplural}{firstpl},%
1660 {text}{text},%
1661 {plural}{plural},%
1662 {description}{desc},%
1663 {descriptionplural}{descplural},%
1664 {symbol}{symbol},%
1665 {symbolplural}{symbolplural},%

```

```

1666 {user1}{useri},%
1667 {user2}{userii},%
1668 {user3}{useriii},%
1669 {user4}{useriv},%
1670 {user5}{userv},%
1671 {user6}{uservi},%
1672 {long}{long},%
1673 {longplural}{longpl},%
1674 {short}{short},%
1675 {shortplural}{shortpl},%
1676 {counter}{counter},%
1677 {parent}{parent}%
1678 }

```

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

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

```
1679 \newcommand*{\@gls@fetchfield}[2]{%
```

Ensure user field name is fully expanded

```
1680 \edef\@gls@thisval{#2}%
```

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

```

1681 \@for\@gls@map:=\@gls@keymap\do{%
1682 \edef\@this@key{\expandafter\@firstoftwo\@gls@map}%
1683 \ifdefequal{\@this@key}{\@gls@thisval}%
1684 {%

```

Found it.

```
1685 \edef#1{\expandafter\@secondoftwo\@gls@map}%
```

Break out of loop.

```

1686 \@endfortrue
1687 }%
1688 {}%
1689 }%
1690 }

```

`\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.

```
1691 \newcommand*{\glsaddkey}{\@ifstar\@sglsaddkey\@glsaddkey}
```

Starred version switches on expansion for this key.

```

1692 \newcommand*{\@sglsaddkey}[1]{%
1693 \key@ifundefined{glossentry}{#1}%
1694 {%

```

```

1695 \expandafter\newcommand\expandafter*\expandafter
1696 {\csname gls@assign@#1@field\endcsname}[2]{%
1697 \@@gls@expand@field{##1}{#1}{##2}%
1698 }%
1699 }%
1700 {}%
1701 \@gls@saddkey{#1}%
1702 }

```

Unstarred version doesn't override default expansion.

```
1703 \newcommand*\@gls@saddkey}[7]{%
```

Check the specified key doesn't already exist.

```

1704 \key@ifundefined{glossentry}{#1}%
1705 {%

```

Set up the key.

```

1706 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1707 \appto\@gls@keymap{,{#1}{#1}}%

```

Set the default value.

```
1708 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
```

Assignment code.

```

1709 \appto\@newglossaryentryposthook{%
1710 \letcs{\@glo@tmp}{@glo@#1}%
1711 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1712 }%

```

Define the no-link commands.

```

1713 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1714 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

1715 \ifcsdef{@gls@user@#1@}%
1716 {%
1717 \PackageError{glossaries}%
1718 {Can't define '\string#5' as helper command
1719 '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1720 {}%
1721 }%
1722 {%
1723 \newrobustcmd*{#5}{\@ifstar{\csuse{@sgls@user@#1}}{\csuse{@gls@user@#1}}}%
1724 \expandafter\newcommand\expandafter*\expandafter
1725 {\csname @sgls@user@#1\endcsname}[1][{%
1726 \csuse{@gls@user@#1}[hyper=false,##1}%
1727 }%
1728 \expandafter\newcommand\expandafter*\expandafter
1729 {\csname @gls@user@#1\endcsname}[2][{%
1730 \new@ifnextchar[%
1731 {\csuse{@gls@user@#1@}{##1}{##2}}%
1732 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%

```

```

1733 \csdef{@gls@user@#1@}##1##2[##3]{%
1734 \@gls@field@link{##1}{##2}{#3{##2}##3}%
1735 }%
1736 }%

```

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

```

1737 \ifcsdef{@Gls@user@#1@}%
1738 {%
1739 \PackageError{glossaries}%
1740 {Can't define '\string#6' as helper command
1741 '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1742 }%
1743 }%
1744 {%
1745 \newrobustcmd*{#6}{\@ifstar{\csuse{@sGls@user@#1}}{\csuse{@Gls@user@#1}}}%
1746 \expandafter\newcommand\expandafter*\expandafter
1747 {\csname @sGls@user@#1\endcsname}[1][{%
1748 \csuse{@Gls@user@#1}[hyper=false,##1]%
1749 }%
1750 \expandafter\newcommand\expandafter*\expandafter
1751 {\csname @Gls@user@#1\endcsname}[2][{%
1752 \new@ifnextchar[%
1753 {\csuse{@Gls@user@#1@}{##1}{##2}}%
1754 {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
1755 \csdef{@Gls@user@#1@}##1##2[##3]{%
1756 \@gls@field@link{##1}{##2}{#4{##2}##3}%
1757 }%
1758 }%

```

Finally the all caps version:

```

1759 \ifcsdef{@GLS@user@#1@}%
1760 {%
1761 \PackageError{glossaries}%
1762 {Can't define '\string#7' as helper command
1763 '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1764 }%
1765 }%
1766 {%
1767 \newrobustcmd*{#7}{\@ifstar{\csuse{@sGLS@user@#1}}{\csuse{@GLS@user@#1}}}%
1768 \expandafter\newcommand\expandafter*\expandafter
1769 {\csname @sGLS@user@#1\endcsname}[1][{%
1770 \csuse{@GLS@user@#1}[hyper=false,##1]%
1771 }%
1772 \expandafter\newcommand\expandafter*\expandafter
1773 {\csname @GLS@user@#1\endcsname}[2][{%
1774 \new@ifnextchar[%
1775 {\csuse{@GLS@user@#1@}{##1}{##2}}%
1776 {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%
1777 \csdef{@GLS@user@#1@}##1##2[##3]{%
1778 \@gls@field@link{##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%

```



```

1779     }%
1780   }%
1781 }%
1782 {%
1783   \PackageError{glossaries}{Key ‘#1’ already exists}{}%
1784 }%
1785 }

```

`\glswritedefhook`

```
1786 \newcommand*{\glswritedefhook}{}

```

`\gls@assign@desc`

```

1787 \newcommand*{\gls@assign@desc}[1]{%
1788   \gls@assign@field{#1}{desc}{\@glo@desc}%
1789   \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1790 }

```

`\longnewglossaryentry`

```

1791 \newcommand{\longnewglossaryentry}[3]{%
1792   \glsdoifnoexists{#1}%
1793   {%
1794     \bgroup
1795     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1796     \long\def\@newglossaryentryprehook{%
1797       \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1798       \@org@newglossaryentryprehook
1799     }%
1800     \renewcommand*{\gls@assign@desc}[1]{%
1801       \global\cslet{glo@glstetoklabel{#1}@desc}{\@glo@desc}%
1802       \global\cslet{glo@glstetoklabel{#1}@descplural}{\@glo@desc}%
1803     }
1804     \gls@defglossaryentry{#1}{#2}%
1805   \egroup
1806 }
1807 }

```

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

```
1808 \@onlypreamble{\longnewglossaryentry}

```

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

```

1809 \newcommand{\longprovideglossaryentry}[3]{%
1810   \ifglstryexists{#1}{}%
1811   {\longnewglossaryentry{#1}{#2}{#3}}%
1812 }
1813 \@onlypreamble{\longprovideglossaryentry}

```

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

Defines a new entry without checking if it already exists.

```
1814 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1815   \edef\@glo@label{\glsdetoklabel{#1}}%
```

Provide a means for user defined keys to reference the label:

```
1816   \let\glslabel\@glo@label
```

Set up defaults. If the name or description keys are omitted, an error will be generated.

```
1817   \let\@glo@name\@gls@name
```

```
1818   \let\@glo@desc\@gls@desc
```

```
1819   \let\@glo@descplural\@gls@default@value
```

```
1820   \let\@glo@type\@gls@default@value
```

```
1821   \let\@glo@symbol\@gls@default@value
```

```
1822   \let\@glo@symbolplural\@gls@default@value
```

```
1823   \let\@glo@text\@gls@default@value
```

```
1824   \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.)

```
1825   \let\@glo@first\@gls@default@value
```

```
1826   \let\@glo@firstplural\@gls@default@value
```

Set the default sort:

```
1827   \let\@glo@sort\@gls@default@value
```

Set the default counter:

```
1828   \let\@glo@counter\@gls@default@value
```

```
1829   \def\@glo@see{}%
```

```
1830   \def\@glo@parent{}%
```

```
1831   \def\@glo@prefix{}%
```

```
1832   \def\@glo@useri{}%
```

```
1833   \def\@glo@userii{}%
```

```
1834   \def\@glo@useriii{}%
```

```
1835   \def\@glo@useriv{}%
```

```
1836   \def\@glo@userv{}%
```

```
1837   \def\@glo@uservi{}%
```

```
1838   \def\@glo@short{}%
```

```
1839   \def\@glo@shortpl{}%
```

```
1840   \def\@glo@long{}%
```

```
1841   \def\@glo@longpl{}%
```

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

```

1842 \newglossaryentryprehook

```

Extract key-val information from third parameter:

```

1843 \setkeys{glossentry}{#2}%

```

Check there is a default glossary.

```

1844 \ifundef\glsdefaulttype
1845 {%
1846   \PackageError{glossaries}%
1847   {No default glossary type (have you used 'nomain'?)}%
1848   {If you use package option 'nomain' you must define
1849    a new glossary before you can define entries}%
1850 }%
1851 {}%

```

Assign type. This must be fully expandable

```

1852 \gls@assign@field{\glsdefaulttype}{\@glo@label}{type}{\@glo@type}%
1853 \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.

```

1854 \ifcsundef{glolist@\@glo@type}%
1855 {%
1856   \PackageError{glossaries}%
1857   {Glossary type '@glo@type' has not been defined}%
1858   {You need to define a new glossary type, before making entries
1859    in it}%
1860 }%
1861 {%
1862   \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
1863   \expandafter\xdef\csname glolist@\@glo@type\endcsname{%
1864     \@glolist@{\@glo@label},}%
1865   }%

```

Initialise level to 0.

```

1866 \gls@level=0\relax

```

Has this entry been assigned a parent?

```

1867 \ifx\@glo@parent\@empty

```

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

```

1868 \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
1869 \else

```

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

```

1870 \ifdefequal\@glo@label\@glo@parent%
1871 {%
1872   \PackageError{glossaries}{Entry '@glo@label' can't be its own parent}{}%
1873   \def\@glo@parent{}%
1874   \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
1875   }%
1876   {%

```

Check the parent exists:

```
1877 \ifglentryexists{\@glo@parent}%
1878 {%
```

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

```
1879 \expandafter\xdef\csname glo@\@glo@label @parent\endcsname{%
1880 \@glo@parent}%
```

Determine level.

```
1881 \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1882 \advance\gls@level by 1\relax
```

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

```
1883 \ifx\@glo@name\@glsnoname
1884 \expandafter\let\expandafter\@glo@name
1885 \csname glo@\@glo@parent @name\endcsname
```

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

```
1886 \ifx\@glo@plural\@gls@default@value
1887 \expandafter\let\expandafter\@glo@plural
1888 \csname glo@\@glo@parent @plural\endcsname
1889 \fi
1890 \fi
1891 }%
1892 {%
```

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

```
1893 \PackageError{glossaries}%
1894 {%
1895 Invalid parent '\@glo@parent'
1896 for entry '\@glo@label' - parent doesn't exist%
1897 }%
1898 {%
1899 Parent entries must be defined before their children%
1900 }%
1901 \def\@glo@parent{%
1902 \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{%
1903 }%
1904 }%
1905 \fi
```

Set the level for this entry

```
1906 \expandafter\xdef\csname glo@\@glo@label @level\endcsname{\number\gls@level}%
```

Define commands associated with this entry:

```
1907 \gls@assign@field{\@glo@name}{\@glo@label}{sortvalue}{\@glo@sort}%
1908 \letcs\@glo@sort{glo@\@glo@label @sortvalue}%
1909 \gls@assign@field{\@glo@name}{\@glo@label}{text}{\@glo@text}%
1910 \expandafter\gls@assign@field\expandafter
1911 {\csname glo@\@glo@label @text\endcsname\glspluralsuffix}%
1912 {\@glo@label}{plural}{\@glo@plural}%
```

```

1913 \expandafter\gls@assign@field\expandafter
1914     {\csname glo@\@glo@label @text\endcsname}%
1915     {\@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.

1916 \ifx\@glo@first\@gls@default@value
1917     \expandafter\gls@assign@field\expandafter
1918     {\csname glo@\@glo@label @plural\endcsname}%
1919     {\@glo@label}{firstpl}{\@glo@firstplural}%
1920 \else
1921     \expandafter\gls@assign@field\expandafter
1922     {\csname glo@\@glo@label @first\endcsname\glspluralsuffix}%
1923     {\@glo@label}{firstpl}{\@glo@firstplural}%
1924 \fi

1925 \ifcsundef{@glotype@\@glo@type @counter}%
1926 {%
1927     \def\@glo@defaultcounter{\@glscounter}%
1928 }%
1929 {%
1930     \letcs\@glo@defaultcounter{@glotype@\@glo@type @counter}%
1931 }%
1932 \gls@assign@field{\@glo@defaultcounter}{\@glo@label}{counter}{\@glo@counter}%
1933 \gls@assign@field{}{\@glo@label}{useri}{\@glo@useri}%
1934 \gls@assign@field{}{\@glo@label}{userii}{\@glo@userii}%
1935 \gls@assign@field{}{\@glo@label}{useriii}{\@glo@useriii}%
1936 \gls@assign@field{}{\@glo@label}{useriv}{\@glo@useriv}%
1937 \gls@assign@field{}{\@glo@label}{userv}{\@glo@userv}%
1938 \gls@assign@field{}{\@glo@label}{uservi}{\@glo@uservi}%
1939 \gls@assign@field{}{\@glo@label}{short}{\@glo@short}%
1940 \gls@assign@field{}{\@glo@label}{shortpl}{\@glo@shortpl}%
1941 \gls@assign@field{}{\@glo@label}{long}{\@glo@long}%
1942 \gls@assign@field{}{\@glo@label}{longpl}{\@glo@longpl}%
1943 \ifx\@glo@name\@gls@noname
1944     \@gls@noname
1945     \let\@glo@name\@gls@default@value
1946 \fi
1947 \gls@assign@field{}{\@glo@label}{name}{\@glo@name}%

```

Set default numberlist if not defined:

```

1948 \ifcsundef{glo@\@glo@label @numberlist}%
1949 {%
1950     \csxdef{glo@\@glo@label @numberlist}{%
1951         \noexpand\@gls@missingnumberlist{\@glo@label}}%
1952 }%
1953 {}%

```

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.

```

1954 \def\@glo@desc{\@glo@first}%
1955 \ifx\@glo@desc\@glo@desc
1956 \let\@glo@desc\@glo@first
1957 \fi
1958 \ifx\@glo@desc\@glsnodesc
1959 \@glsnodesc
1960 \let\@glo@desc\@gls@default@value
1961 \fi
1962 \gls@assign@desc{\@glo@label}%

```

Set the sort key for this entry:

```

1963 \@gls@defsort{\@glo@type}{\@glo@label}%

1964 \def\@glo@symbol{\@glo@text}%
1965 \ifx\@glo@symbol\@glo@symbol
1966 \let\@glo@symbol\@glo@text
1967 \fi
1968 \gls@assign@field{\relax}{\@glo@label}{symbol}{\@glo@symbol}%
1969 \expandafter
1970 \gls@assign@field\expandafter
1971 {\csname glo@\@glo@label @symbol\endcsname}
1972 {\@glo@label}{symbolplural}{\@glo@symbolplural}%

```

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

```

1973 \expandafter\xdef\csname glo@\@glo@label @flagfalse\endcsname{%
1974 \noexpand\global
1975 \noexpand\let\expandafter\noexpand
1976 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iffalse
1977 }%
1978 \expandafter\xdef\csname glo@\@glo@label @flagtrue\endcsname{%
1979 \noexpand\global
1980 \noexpand\let\expandafter\noexpand
1981 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iftrue
1982 }%
1983 \csname glo@\@glo@label @flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1984 \ifx\@glo@see\@empty
1985 \else
1986 \protected@edef\@do@glsee{%
1987 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1988 \noexpand\@nil
1989 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{\@glo@label}}%
1990 \@do@glsee
1991 \fi

```

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

```

1992 \do@glo@storeentry{\@glo@label}%

```

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

```

1993 \newglossaryentryposthook
1994 }

glossaryentryprehook Allow extra information to be added to glossary entries:
1995 \newcommand*{\@newglossaryentryprehook}{}

glossaryentryposthook Allow extra information to be added to glossary entries:
1996 \newcommand*{\@newglossaryentryposthook}{}

\glsmoveentry Moves entry whose label is given by first argument to the glossary named in the
second argument.
1997 \newcommand*{\glsmoveentry}[2]{%
1998 \edef\@glo@thislabel{\glsdetoklabel{#1}}%
1999 \edef\glo@type{\csname glo@\@glo@thislabel @type\endcsname}%
2000 \def\glo@list{,%}
2001 \forglsentries[\glo@type]{\glo@label}%
2002 {%

2003 \ifdefequal\@glo@thislabel\glo@label
2004 {}{\eappto\glo@list{\glo@label,%}%
2005 }%
2006 \cslet\glo@list@\glo@type{\glo@list}%
2007 \csdef\glo@\@glo@thislabel @type{#2}%
2008 }

@glossaryentryfield Indicate what command should be used to display each entry in the glossary.
(This enables the glossaries-accsupp package to use \accsuppglossaryentryfield
instead.)
2009 \ifglxindy
2010 \newcommand*{\@glossaryentryfield}{\string\glossentry}
2011 \else
2012 \newcommand*{\@glossaryentryfield}{\string\glossentry}
2013 \fi

glossarysubentryfield Indicate what command should be used to display each subentry in the glos-
sary. (This enables the glossaries-accsupp package to use \accsuppglossarysubentryfield
instead.)
2014 \ifglxindy
2015 \newcommand*{\@glossarysubentryfield}{%
2016 \string\subglossentry}
2017 \else
2018 \newcommand*{\@glossarysubentryfield}{%
2019 \string\subglossentry}
2020 \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>@entry, where <label> is the entry's label. (This doesn't include any formatting or location information.)

```
2021 \newcommand{\@glo@storeentry}[1]{%
```

Escape makeindex/xindy special characters in the label:

```
2022 \edef\@glo@esclabel{#1}%
```

```
2023 \@gls@checkmkidxchars\@glo@esclabel
```

Get the sort string and escape any special characters

```
2024 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
```

```
2025 \@gls@checkmkidxchars\@glo@sort
```

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

```
2026 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
2027 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
2028 \ifglsxindy
```

Store using xindy syntax.

```
2029 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```
2030 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
```

```
2031 (\string"\@glo@sort\string" %
```

```
2032 \string"\@glo@prefix\@glossaryentryfield{\@glo@esclabel}\string") %
```

```
2033 }%
```

```
2034 \else
```

Entry has a parent

```
2035 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
```

```
2036 \csname glo@\@glo@parent @index\endcsname
```

```
2037 (\string"\@glo@sort\string" %
```

```
2038 \string"\@glo@prefix\@glossarysubentryfield
```

```
2039 {\csname glo@#1@level\endcsname}{\@glo@esclabel}\string") %
```

```
2040 }%
```

```
2041 \fi
```

```
2042 \else
```

Store using makeindex syntax.

```
2043 \ifx\@glo@parent\@empty
```

Sanitize \@glo@prefix

```
2044 \@onelevel@sanitize\@glo@prefix
```

Entry doesn't have a parent

```
2045 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
```

```
2046 \@glo@sort\@gls@actualchar\@glo@prefix
```

```
2047 \@glossaryentryfield{\@glo@esclabel}%
```



```

2048     }%
2049     \else
      Entry has a parent
2050     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2051       \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2052       \@glo@sort\@gls@actualchar\@glo@prefix
2053       \@glossarysubentryfield
2054       {\csname glo@#1@level\endcsname}\@glo@esclabel}%
2055     }%
2056     \fi
2057   \fi
2058 }

```

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`

```

2059 \AtBeginDocument{%
2060   \ifpackageloaded{amsmath}%
2061   {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2062   {}}%
2063 }
2064 \newcommand*{\@gls@ifnotmeasuring}[1]{%
2065   \ifmeasuring@
2066   \else
2067     #1%
2068   \fi
2069 }
2070 \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.

```

2071 \newcommand*{\glsreset}[1]{%
2072   \gls@ifnotmeasuring
2073   {%
2074     \glsdoifexists{#1}%
2075     {%
2076       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagfalse\endcsname
2077     }%
2078   }%
2079 }

```

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

```
2080 \newcommand*{\glslocalreset}[1]{%
2081   \gls@ifnotmeasuring
2082   {%
2083     \glsdoifexists{#1}%
2084     {%
2085       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iffalse
2086     }%
2087   }%
2088 }
```

`\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.

```
2089 \newcommand*{\glsunset}[1]{%
2090   \gls@ifnotmeasuring
2091   {%
2092     \glsdoifexists{#1}%
2093     {%
2094       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagtrue\endcsname
2095     }%
2096   }%
2097 }
```

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

```
2098 \newcommand*{\glslocalunset}[1]{%
2099   \gls@ifnotmeasuring
2100   {%
2101     \glsdoifexists{#1}%
2102     {%
2103       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iftrue
2104     }%
2105   }%
2106 }
```

Reset all entries for the named glossaries (supplied in a comma-separated list).

Syntax: `\glsresetall[<glossary-list>]`

`\glsresetall`

```
2107 \newcommand*{\glsresetall}[1][\@glo@types]{%
2108   \forallglsentries[#1]{\@glsentry}%
2109   {%
2110     \glsreset{\@glsentry}%
2111   }%
2112 }
```

As above, but with only a local effect:

`\glslocalresetall`

```
2113 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
```

```

2114 \foralllglentries[#1]{\@glentry}%
2115 {%
2116     \glslocalreset{\@glentry}%
2117 }%
2118 }

```

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

```

\glsunsetall
2119 \newcommand*\glsunsetall}[1][\@glo@types]{%
2120     \foralllglentries[#1]{\@glentry}%
2121     {%
2122         \glsunset{\@glentry}%
2123     }%
2124 }

```

As above, but with only a local effect:

```

\glslocalunsetall
2125 \newcommand*\glslocalunsetall}[1][\@glo@types]{%
2126     \foralllglentries[#1]{\@glentry}%
2127     {%
2128         \glslocalunset{\@glentry}%
2129     }%
2130 }

```

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

`\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
2131 \newcommand*\loadglsentries}[2][\@gls@default]{%
2132     \let\@gls@default\glsdefaulttype
2133     \def\glsdefaulttype{#1}\input{#2}%
2134     \let\glsdefaulttype\@gls@default
2135 }

```

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

`\loadglsentries` can only be used in the preamble:
2136 `\@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`

2137 `\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

2138 `\newcommand*{\glsentryfmt}{%`
2139 `\@@gls@default@entryfmt\glsdisplayfirst\glsdisplay`
2140 `}`

Format that provides backwards compatibility:

2141 `\newcommand*{\@@gls@default@entryfmt}[2]{%`
2142 `\ifdefempty\glscustomtext`
2143 `{%`
2144 `\glsifplural`
2145 `{%`

Plural form

2146 `\glscapscase`
2147 `{%`

Don’t adjust case

2148 `\ifglsused\glslabel`
2149 `{%`

Subsequent use

2150 `#2{\glsentryplural{\glslabel}}%`
2151 `{\glsentrydescplural{\glslabel}}%`
2152 `{\glsentrysymbolplural{\glslabel}}{\glsinsert}}%`
2153 `}%`
2154 `{%`

First use

2155 `#1{\glsentryfirstplural{\glslabel}}%`
2156 `{\glsentrydescplural{\glslabel}}%`
2157 `{\glsentrysymbolplural{\glslabel}}{\glsinsert}}%`

```

2158     }%
2159     }%
2160     {%

```

Make first letter upper case

```

2161     \ifglused\glslabel
2162     {%

```

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 `\defglentryfmt`, which avoids the issues caused by fragile commands.)

```

2163     \ifbool{glscompatible-3.07}%
2164     {%
2165         \protected@edef\@glo@etext{%
2166             #2{\glentryplural{\glslabel}}%
2167             {\glentrydescplural{\glslabel}}%
2168             {\glentrysymbolplural{\glslabel}}{\glsinsert}}%
2169         \xmakefirstuc\@glo@etext
2170     }%
2171     {%
2172         #2{\Glsentryplural{\glslabel}}%
2173         {\glentrydescplural{\glslabel}}%
2174         {\glentrysymbolplural{\glslabel}}{\glsinsert}%
2175     }%
2176     }%
2177     {%

```

First use

```

2178     \ifbool{glscompatible-3.07}%
2179     {%
2180         \protected@edef\@glo@etext{%
2181             #1{\glentryfirstplural{\glslabel}}%
2182             {\glentrydescplural{\glslabel}}%
2183             {\glentrysymbolplural{\glslabel}}{\glsinsert}}%
2184         \xmakefirstuc\@glo@etext
2185     }%
2186     {%
2187         #1{\Glsentryfirstplural{\glslabel}}%
2188         {\glentrydescplural{\glslabel}}%
2189         {\glentrysymbolplural{\glslabel}}{\glsinsert}%
2190     }%
2191     }%
2192     }%
2193     {%

```

Make all upper case

```

2194     \ifglused\glslabel
2195     {%

```

Subsequent use

```

2196      \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}}%
2197      {\glsentrydescplural{\glslabel}}}%
2198      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2199      }%
2200      {%

```

First use

```

2201      \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}}%
2202      {\glsentrydescplural{\glslabel}}}%
2203      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2204      }%
2205      }%
2206      }%
2207      {%

```

Singular form

```

2208      \glscapscase
2209      {%

```

Don't adjust case

```

2210      \ifglused\glslabel
2211      {%

```

Subsequent use

```

2212      #2{\glsentrytext{\glslabel}}}%
2213      {\glsentrydesc{\glslabel}}}%
2214      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2215      }%
2216      {%

```

First use

```

2217      #1{\glsentryfirst{\glslabel}}}%
2218      {\glsentrydesc{\glslabel}}}%
2219      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2220      }%
2221      }%
2222      {%

```

Make first letter upper case

```

2223      \ifglused\glslabel
2224      {%

```

Subsequent use

```

2225      \ifbool{glscompatible-3.07}%
2226      {%
2227      \protected@edef\@glo@etext{%
2228      #2{\glsentrytext{\glslabel}}}%
2229      {\glsentrydesc{\glslabel}}}%
2230      {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2231      \xmakefirstuc\@glo@etext
2232      }%
2233      {%

```

```

2234         #2{\Glsentrytext{\glslabel}}%
2235         {\glsentrydesc{\glslabel}}%
2236         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2237     }%
2238 }%
2239 {%

```

First use

```

2240     \ifbool{glscompatible-3.07}%
2241     {%
2242         \protected@edef\@glo@etext{%
2243             #1{\glsentryfirst{\glslabel}}%
2244             {\glsentrydesc{\glslabel}}%
2245             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2246         \xmakefirstuc\@glo@etext
2247     }%
2248     {%
2249         #1{\Glsentryfirst{\glslabel}}%
2250         {\glsentrydesc{\glslabel}}%
2251         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2252     }%
2253 }%
2254 }%
2255 {%

```

Make all upper case

```

2256     \ifglsused\glslabel
2257     {%

```

Subsequent use

```

2258         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}%
2259         {\glsentrydesc{\glslabel}}%
2260         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2261     }%
2262     {%

```

First use

```

2263         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}%
2264         {\glsentrydesc{\glslabel}}%
2265         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2266     }%
2267 }%
2268 }%
2269 }%
2270 {%

```

Custom text provided in \glsdisp

```

2271     \ifglsused{\glslabel}%
2272     {%

```

Subsequent use

```

2273         #2{\glscustomtext}%

```

```

2274      {\glsentrydesc{\glslabel}}}%
2275      {\glsentrysymbol{\glslabel}}{}}%
2276    }%
2277    {%

```

First use

```

2278      #1{\glscustomtext}%
2279      {\glsentrydesc{\glslabel}}}%
2280      {\glsentrysymbol{\glslabel}}{}}%
2281    }%
2282    }%
2283 }

```

`\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.

```

2284 \newcommand*{\glsgenentryfmt}{%
2285   \ifdefempty\glscustomtext
2286   {%
2287     \glsifplural
2288     {%

```

Plural form

```

2289     \glscapscase
2290     {%

```

Don't adjust case

```

2291     \ifglsused\glslabel
2292     {%

```

Subsequent use

```

2293     \glsentryplural{\glslabel}\glsinsert
2294     }%
2295     {%

```

First use

```

2296     \glsentryfirstplural{\glslabel}\glsinsert
2297     }%
2298     }%
2299     {%

```

Make first letter upper case

```

2300     \ifglsused\glslabel
2301     {%

```

Subsequent use.

```

2302     \Glsentryplural{\glslabel}\glsinsert
2303     }%
2304     {%

```

First use

```

2305     \Glsentryfirstplural{\glslabel}\glsinsert
2306     }%

```



```

2307     }%
2308     {%

    Make all upper case
2309         \ifglused\glslabel
2310         {%

    Subsequent use
2311         \mfirstucMakeUppercase
2312         {\glentryplural{\glslabel}\glsinsert}%
2313     }%
2314     {%

    First use
2315         \mfirstucMakeUppercase
2316         {\glentryfirstplural{\glslabel}\glsinsert}%
2317     }%
2318     }%
2319     }%
2320     {%

    Singular form
2321     \glscapscase
2322     {%

    Don't adjust case
2323     \ifglused\glslabel
2324     {%

    Subsequent use
2325     \glentrytext{\glslabel}\glsinsert
2326     }%
2327     {%

    First use
2328     \glentryfirst{\glslabel}\glsinsert
2329     }%
2330     }%
2331     {%

    Make first letter upper case
2332     \ifglused\glslabel
2333     {%

    Subsequent use
2334     \Glentrytext{\glslabel}\glsinsert
2335     }%
2336     {%

    First use
2337     \Glentryfirst{\glslabel}\glsinsert
2338     }%
2339     }%
2340     {%

```

Make all upper case

```
2341      \ifglsused\glslabel
2342      {%
```

Subsequent use

```
2343      \mfirstucMakeUppercase{\glsentrytext{\glslabel}\glsinsert}%
2344      }%
2345      {%
```

First use

```
2346      \mfirstucMakeUppercase{\glsentryfirst{\glslabel}\glsinsert}%
2347      }%
2348      }%
2349      }%
2350      }%
2351      {%
```

Custom text provided in \glsdisp. (The insert is most likely to be empty at this point.)

```
2352      \glscustomtext\glsinsert
2353      }%
2354 }
```

`\glsngenacfmt` Define a generic acronym format that uses the long and short keys (or their plurals) and `\acrfullformat`, `\firstacronymfont` and `\acronymfont`.

```
2355 \newcommand*{\glsngenacfmt}{%
2356   \ifdefempty\glscustomtext
2357   {%
2358     \ifglsused\glslabel
2359     {%
```

Subsequent use:

```
2360     \glsifplural
2361     {%
```

Subsequent plural form:

```
2362     \glscapscase
2363     {%
```

Subsequent plural form, don't adjust case:

```
2364     \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2365     }%
2366     {%
```

Subsequent plural form, make first letter upper case:

```
2367     \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2368     }%
2369     {%
```

Subsequent plural form, all caps:

```
2370     \mfirstucMakeUppercase
2371     {\acronymfont{\glsentryshortpl{\glslabel}}\glsinsert}%
```

2372 }%
 2373 }%
 2374 {%

Subsequent singular form

2375 \glscapscase
 2376 {%

Subsequent singular form, don't adjust case:

2377 \acronymfont{\glstryshort{\glslabel}}\glsinsert
 2378 }%
 2379 {%

Subsequent singular form, make first letter upper case:

2380 \acronymfont{\Glsstryshort{\glslabel}}\glsinsert
 2381 }%
 2382 {%

Subsequent singular form, all caps:

2383 \mfirstucMakeUppercase
 2384 {\acronymfont{\glstryshort{\glslabel}}\glsinsert}%
 2385 }%
 2386 }%
 2387 }%
 2388 {%

First use:

2389 \glsifplural
 2390 {%

First use plural form:

2391 \glscapscase
 2392 {%

First use plural form, don't adjust case:

2393 \genplacrfullformat{\glslabel}{\glsinsert}%
 2394 }%
 2395 {%

First use plural form, make first letter upper case:

2396 \Genplacrfullformat{\glslabel}{\glsinsert}%
 2397 }%
 2398 {%

First use plural form, all caps:

2399 \mfirstucMakeUppercase
 2400 {\genplacrfullformat{\glslabel}{\glsinsert}}%
 2401 }%
 2402 }%
 2403 {%

First use singular form

2404 \glscapscase
 2405 {%

First use singular form, don't adjust case:

```
2406      \genacrfullformat{\glslabel}{\glsinsert}%
2407      }%
2408      {%
```

First use singular form, make first letter upper case:

```
2409      \Genacrfullformat{\glslabel}{\glsinsert}%
2410      }%
2411      {%
```

First use singular form, all caps:

```
2412      \mfirstucMakeUppercase
2413      {\genacrfullformat{\glslabel}{\glsinsert}}%
2414      }%
2415      }%
2416      }%
2417      }%
2418      {%
```

User supplied text.

```
2419      \glscustomtext
2420      }%
2421 }
```

`\genacrfullformat` `\genacrfullformat{<label>}{<insert>}`

The full format used by `\glsngenacfmt` (singular).

```
2422 \newcommand*{\genacrfullformat}[2]{%
2423   \glsentrylong{#1}#2\space
2424   (\protect\firstacronymfont{\glsentryshort{#1}})%
2425 }
```

`\Genacrfullformat` `\Genacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2426 \newcommand*{\Genacrfullformat}[2]{%
2427   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
2428   \xmakefirstuc\gls@text
2429 }
```

`\genplacrfullformat` `\genplacrfullformat{<label>}{<insert>}`

The full format used by `\glsngenacfmt` (plural).

```
2430 \newcommand*{\genplacrfullformat}[2]{%
2431   \glsentrylongpl{#1}#2\space
2432   (\protect\firstacronymfont{\glsentryshortpl{#1}})%
```

2433 }

`\Genplacrfullformat` `\Genplacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2434 \newcommand*{\Genplacrfullformat}[2]{%
2435   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2436   \xmakefirstuc\gls@text
2437 }
```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2438 \newcommand*{\glsdisplayfirst}[4]{#1#4}
```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2439 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2440 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2441   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2442   Use \string\defglsentryfmt\space instead}%
2443   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2444   \edef\@gls@doentrydef{%
2445     \noexpand\defglsentryfmt[#1]{%
2446       \noexpand\ifcsdef{gls@#1@displayfirst}%
2447       {%
2448         \noexpand\@gls@default@entryfmt
2449         {\noexpand\csuse{gls@#1@displayfirst}}
2450         {\noexpand\csuse{gls@#1@display}}}%
2451       }%
2452       {%
2453         \noexpand\@gls@default@entryfmt
2454         {\noexpand\glsdisplayfirst}
2455         {\noexpand\csuse{gls@#1@display}}}%
2456       }%
2457     }%
2458   }%
2459   \@gls@doentrydef
2460 }
```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2461 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2462   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2463   Use \string\defglsentryfmt\space instead}%
2464   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2465   \edef\@gls@doentrydef{%
2466     \noexpand\defglsentryfmt[#1]{%
2467       \noexpand\ifcsdef{gls@#1@display}%

```

```

2468      {%
2469      \noexpand\@@gls@default@entryfmt
2470      {\noexpand\csuse{gls@#1@displayfirst}}
2471      {\noexpand\csuse{gls@#1@display}}}%
2472    }%
2473    {%
2474    \noexpand\@@gls@default@entryfmt
2475    {\noexpand\csuse{gls@#1@displayfirst}}}%
2476    {\noexpand\glsdisplay}
2477    }%
2478  }%
2479 }%
2480 \@gls@doentrydef
2481 }

```

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 \TeX 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.

```

2482 \define@key{glslink}{counter}{%
2483   \ifcsundef{c@#1}%
2484   {%
2485     \PackageError{glossaries}%
2486     {There is no counter called ‘#1’}%
2487     {%
2488       The counter key should have the name of a valid counter
2489       as its value%
2490     }%
2491   }%
2492   {%
2493     \def\@gls@counter{#1}%
2494   }%
2495 }

```

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.

```
2496 \define@key{glslink}{format}{%
2497   \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.

```
2498 \define@boolkey{glslink}{hyper}[true]{}
```

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.

```
2499 \define@boolkey{glslink}{local}[true]{}
```

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 whether or not we are using the starred version:

```
\glslink
```

```
2500 \newrobustcmd*{\glslink}{%
2501   \@ifstar\@sgls@link\@gls@@link
2502 }
```

`\@sgls@link` The starred version of `\glslink` calls the unstarred version with hyperlinks disabled.

```
2503 \newcommand*{\@sgls@link}[1][]{\@gls@@link[hyper=false,#1]}
```

`\@gls@@link` The unstarred version of `\glslink` checks for the existence of the term. 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.

```
2504 \newcommand*{\@gls@@link}[3][]{%
2505   \ifglsentryexists{#2}%
2506   {%
2507     \@gls@link[#1]{#2}{#3}%
2508   }{%
2509     \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2510       defined}{You need to define a glossary entry before you
2511       can use it.}%

```

Display the specified text. (The entry doesn't exist so there's nothing to link it to.)

```
2512 \glstextformat{#3}%
2513 }%
2514 }
```

`\@gls@link`

```
2515 \def\@gls@link[#1]#2#3{%
    Inserting \leavevmode suggested by Donald Arseneau (avoids problem with
    tabularx).
```

```
2516 \leavevmode
2517 \edef\glslabel{\glsdetoklabel{#2}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
2518 \def\@gls@link@opts{#1}%
2519 \let\@gls@link@label\glslabel
2520 \def\@glsnumberformat{\glsnumberformat}%
2521 \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
    If this is in one of the “nohypertypes” glossaries, suppress the hyperlink by default
```

```
2522 \edef\gls@type{\csname glo@\glslabel @type\endcsname}%
2523 \expandafter\DTLifinlist\expandafter
2524 {\gls@type}{\@gls@nohyperlist}%
2525 {%
2526 \KV@glslink@hyperfalse
2527 }%
2528 {%
2529 \KV@glslink@hypertrue
2530 }%
2531 \setkeys{glslink}{#1}%
    Store the entry's counter in \theglsentrycounter
```

```
2532 \@gls@saveentrycounter
    Define sort key if necessary:
```

```
2533 \@gls@setsort{\glslabel}%
    (De-tok'ing done by \@do@wrglossary)
2534 \@do@wrglossary{#2}%
2535 \ifKV@glslink@hyper
2536 \@glslink{\glslinkprefix\glslabel}{\glstextformat{#3}}%
2537 \else
2538 \glstextformat{#3}%
2539 \fi
2540 }
```

`\glslinkprefix`

```
2541 \newcommand*{\glslinkprefix}{glo:}
```


`\glentrycounter` Set default value of entry counter

```

2542 \def\glentrycounter{\glscounter}%

```

`\glssaveentrycounter` Need to check if using equation counter in align environment:

```

2543 \newcommand*{\@gls@saveentrycounter}{%
2544   \def\@gls@Hcounter{}%

   Are we using equation counter?
2545   \ifthenelse{\equal{\@gls@counter}{equation}}{%
2546     {

       If we in align environment, \xatlevel@ will be defined. (Can't test for \@currentvir
       as may be inside an inner environment.)
2547     \ifcsundef{xatlevel@}%
2548     {%
2549       \edef\theglentrycounter{\expandafter\noexpand
2550         \csname the\@gls@counter\endcsname}%
2551     }%
2552     {%
2553       \ifx\xatlevel@\@empty
2554         \edef\theglentrycounter{\expandafter\noexpand
2555           \csname the\@gls@counter\endcsname}%
2556       \else
2557         \savecounters@
2558         \advance\c@equation by 1\relax
2559         \edef\theglentrycounter{\csname the\@gls@counter\endcsname}%

       Check if hyperref version of this counter
2560         \ifcsundef{theH\@gls@counter}%
2561         {%
2562           \def\@gls@Hcounter{\theglentrycounter}%
2563         }%
2564         {%
2565           \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2566         }%
2567         \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2568         \restorecounters@
2569       \fi
2570     }%
2571   }%
2572   {%

       Not using equation counter so no special measures:
2573     \edef\theglentrycounter{\expandafter\noexpand
2574       \csname the\@gls@counter\endcsname}%
2575   }%

       Check if hyperref version of this counter
2576   \ifx\@gls@Hcounter\@empty
2577     \ifcsundef{theH\@gls@counter}%
2578     {%

```

```

2579     \def\theHglentrycounter{\theHglentrycounter}%
2580 }%
2581 {%
2582     \protected@edef\theHglentrycounter{\expandafter\noexpand
2583         \csname theH\@gls@counter\endcsname}%
2584 }%
2585 \fi
2586 }

```

`\@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.

```

2587 \def\@set@glo@numformat#1#2#3#4{%
2588     \expandafter\@glo@check@mkidxrangechar#3\@nil
2589     \protected@edef#1{%
2590         \@glo@prefix setentrycounter[#4]{#2}%
2591         \expandafter\string\csname\@glo@suffix\endcsname
2592     }%
2593     \@gls@checkmkidxchars#1%
2594 }

```

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.

```

2595 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2596 \if#1(\relax
2597     \def\@glo@prefix{(%}%
2598     \if\relax#2\relax
2599         \def\@glo@suffix{glsnumberformat}%
2600     \else
2601         \def\@glo@suffix{#2}%
2602     \fi
2603 \else
2604     \if#1)\relax
2605         \def\@glo@prefix{)%}%
2606         \if\relax#2\relax
2607             \def\@glo@suffix{glsnumberformat}%
2608         \else
2609             \def\@glo@suffix{#2}%
2610         \fi
2611     \else
2612         \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2613     \fi
2614 \fi}

```

`\@gls@escbsdq` Escape backslashes and double quote marks. The argument must be a control sequence.

```

2615 \newcommand*{\@gls@escbsdq}[1]{%
2616   \def\@gls@checkedmkidx{}%
2617   \let\gls@xdystring=#1\relax
2618   \@onelevel@sanitize\gls@xdystring
2619   \edef\do@gls@xdycheckbackslash{%
2620     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2621     \@backslashchar\@backslashchar\noexpand\@null}%
2622   \do@gls@xdycheckbackslash
2623   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
2624   \def\@gls@checkedmkidx{}%
2625   \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
2626   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%

```

Unsanitize `\gls@numberpage`, `\gls@alphpage`, `\gls@Alphpage` and `\gls@romanpage` (thanks to David Carlisle for the suggestion.)

```

2627   \@for\@gls@tmp:=\gls@protected@pagefmts\do
2628   {%
2629     \edef\@gls@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gls@tmp}%
2630     \@onelevel@sanitize\@gls@sanitized@tmp
2631     \edef\gls@dostubst{%
2632       \noexpand\DTLsubstituteall\noexpand\gls@xdystring
2633       {\@gls@sanitized@tmp}{\expandonce\@gls@tmp}%
2634     }%
2635     \gls@dostubst
2636   }%

```

Assign to required control sequence

```

2637   \let#1=\gls@xdystring
2638 }

```

Catch special characters (argument must be a control sequence):

`\gls@checkmkidxchars`

```

2639 \newcommand{\@gls@checkmkidxchars}[1]{%
2640   \ifglxindy
2641     \@gls@escbsdq{#1}%
2642   \else
2643     \def\@gls@checkedmkidx{}%
2644     \expandafter\@gls@checkquote#1\@nil""\null
2645     \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2646     \def\@gls@checkedmkidx{}%
2647     \expandafter\@gls@checkescquote#1\@nil\""\null
2648     \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2649     \def\@gls@checkedmkidx{}%
2650     \expandafter\@gls@checkescactual#1\@nil\?\?\null
2651     \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2652     \def\@gls@checkedmkidx{}%
2653     \expandafter\@gls@checkactual#1\@nil??\null

```

```

2654 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2655 \def\@gls@checkedmkidx{}%
2656 \expandafter\@gls@checkbar#1\@nil||\null
2657 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2658 \def\@gls@checkedmkidx{}%
2659 \expandafter\@gls@checkesbar#1\@nil|||\null
2660 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2661 \def\@gls@checkedmkidx{}%
2662 \expandafter\@gls@checklevel#1\@nil!!\null
2663 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2664 \fi
2665 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

2666 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

2667 \newtoks\@gls@tmpb

```

\@gls@checkquote Replace " with "" since " is a makeindex special character.

```

2668 \def\@gls@checkquote#1"#2"#3\null{%
2669 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2670 \toks@={#1}%
2671 \ifx\null#2\null
2672 \ifx\null#3\null
2673 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2674 \def\@gls@checkquote{\relax}%
2675 \else
2676 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2677 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2678 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2679 \fi
2680 \else
2681 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2682 \@gls@quotechar\@gls@quotechar}%
2683 \ifx\null#3\null
2684 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2685 \else
2686 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2687 \fi
2688 \fi
2689 \@gls@checkquote
2690 }

```

\@gls@checkescquote Do the same for \":

```

2691 \def\@gls@checkescquote#1\"#2\"#3\null{%
2692 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%

```

```

2693 \toks@={#1}%
2694 \ifx\null#2\null
2695 \ifx\null#3\null
2696 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2697 \def\@@gls@checkescquote{\relax}%
2698 \else
2699 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2700 \@gls@quotechar\string"\@gls@quotechar
2701 \@gls@quotechar\string"\@gls@quotechar}%
2702 \def\@@gls@checkescquote{\@gls@checkescquote#3\null}%
2703 \fi
2704 \else
2705 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2706 \@gls@quotechar\string"\@gls@quotechar}%
2707 \ifx\null#3\null
2708 \def\@@gls@checkescquote{\@gls@checkescquote#2\""\null}%
2709 \else
2710 \def\@@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
2711 \fi
2712 \fi
2713 \@@gls@checkescquote
2714 }

```

@gls@checkescactual Similarly for \? (which is replaces @ as makeindex's special character):

```

2715 \def\@gls@checkescactual#1\?#2\?#3\null{%
2716 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2717 \toks@={#1}%
2718 \ifx\null#2\null
2719 \ifx\null#3\null
2720 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2721 \def\@@gls@checkescactual{\relax}%
2722 \else
2723 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2724 \@gls@quotechar\string"\@gls@actualchar
2725 \@gls@quotechar\string"\@gls@actualchar}%
2726 \def\@@gls@checkescactual{\@gls@checkescactual#3\null}%
2727 \fi
2728 \else
2729 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2730 \@gls@quotechar\string"\@gls@actualchar}%
2731 \ifx\null#3\null
2732 \def\@@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2733 \else
2734 \def\@@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2735 \fi
2736 \fi
2737 \@@gls@checkescactual
2738 }

```

\@gls@checkescbar Similarly for \|:

```
2739 \def\@gls@checkescbar#1\|#2\|#3\null{%
2740   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2741   \toks@={#1}%
2742   \ifx\null#2\null
2743     \ifx\null#3\null
2744       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2745       \def\@gls@checkescbar{\relax}%
2746     \else
2747       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2748         \@gls@quotechar\string\\"\@gls@encapchar
2749         \@gls@quotechar\string\\"\@gls@encapchar}%
2750       \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
2751     \fi
2752   \else
2753     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2754       \@gls@quotechar\string\\"\@gls@encapchar}%
2755     \ifx\null#3\null
2756       \def\@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2757     \else
2758       \def\@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2759     \fi
2760   \fi
2761 \@gls@checkescbar
2762 }
```

\@gls@checkesclevel Similarly for \!:

```
2763 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2764   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2765   \toks@={#1}%
2766   \ifx\null#2\null
2767     \ifx\null#3\null
2768       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2769       \def\@gls@checkesclevel{\relax}%
2770     \else
2771       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2772         \@gls@quotechar\string\!\@gls@levelchar
2773         \@gls@quotechar\string\!\@gls@levelchar}%
2774       \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2775     \fi
2776   \else
2777     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2778       \@gls@quotechar\string\!\@gls@levelchar}%
2779     \ifx\null#3\null
2780       \def\@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2781     \else
2782       \def\@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2783     \fi
2784   \fi
```

```

2785 \@@gls@checkesclevel
2786 }

```

\@gls@checkbar and for |:

```

2787 \def\@gls@checkbar#1|#2|#3\null{%
2788   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2789   \toks@={#1}%
2790   \ifx\null#2\null
2791     \ifx\null#3\null
2792       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2793       \def\@@gls@checkbar{\relax}%
2794     \else
2795       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2796         \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
2797       \def\@@gls@checkbar{\@gls@checkbar#3\null}%
2798     \fi
2799   \else
2800     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2801       \@gls@quotechar\@gls@encapchar}%
2802     \ifx\null#3\null
2803       \def\@@gls@checkbar{\@gls@checkbar#2||\null}%
2804     \else
2805       \def\@@gls@checkbar{\@gls@checkbar#2|#3\null}%
2806     \fi
2807   \fi
2808   \@gls@checkbar
2809 }

```

\@gls@checklevel and for !:

```

2810 \def\@gls@checklevel#1!#2!#3\null{%
2811   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2812   \toks@={#1}%
2813   \ifx\null#2\null
2814     \ifx\null#3\null
2815       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2816       \def\@@gls@checklevel{\relax}%
2817     \else
2818       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2819         \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
2820       \def\@@gls@checklevel{\@gls@checklevel#3\null}%
2821     \fi
2822   \else
2823     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2824       \@gls@quotechar\@gls@levelchar}%
2825     \ifx\null#3\null
2826       \def\@@gls@checklevel{\@gls@checklevel#2!!\null}%
2827     \else
2828       \def\@@gls@checklevel{\@gls@checklevel#2!#3\null}%
2829     \fi

```

```

2830 \fi
2831 \@@gls@checklevel
2832 }

```

\@gls@checkactual and for ?:

```

2833 \def\@gls@checkactual#1?#2?#3\null{%
2834 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2835 \toks@={#1}%
2836 \ifx\null#2\null
2837 \ifx\null#3\null
2838 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2839 \def\@@gls@checkactual{\relax}%
2840 \else
2841 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2842 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
2843 \def\@@gls@checkactual{\@gls@checkactual#3\null}%
2844 \fi
2845 \else
2846 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2847 \@gls@quotechar\@gls@actualchar}%
2848 \ifx\null#3\null
2849 \def\@@gls@checkactual{\@gls@checkactual#2??\null}%
2850 \else
2851 \def\@@gls@checkactual{\@gls@checkactual#2?#3\null}%
2852 \fi
2853 \fi
2854 \@@gls@checkactual
2855 }

```

\@gls@xdycheckquote As before but for use with xindy

```

2856 \def\@gls@xdycheckquote#1"#2"#3\null{%
2857 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2858 \toks@={#1}%
2859 \ifx\null#2\null
2860 \ifx\null#3\null
2861 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2862 \def\@@gls@xdycheckquote{\relax}%
2863 \else
2864 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2865 \string\string}%
2866 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
2867 \fi
2868 \else
2869 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2870 \string\}%
2871 \ifx\null#3\null
2872 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
2873 \else
2874 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%

```



```

2875     \fi
2876     \fi
2877     \@gls@xdycheckquote
2878 }

```

s@xdycheckbackslash Need to escape all backslashes for xindy. Define command that will define

```

\@gls@xdycheckbackslash
2879 \edef\def@gls@xdycheckbackslash{%
2880 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
2881 ##2\@backslashchar##3\noexpand\null{%
2882 \noexpand\@gls@tmpb=\noexpand\expandafter
2883 {\noexpand\@gls@checkedmkidx}%
2884 \noexpand\toks@={##1}%
2885 \noexpand\ifx\noexpand\null##2\noexpand\null
2886 \noexpand\ifx\noexpand\null##3\noexpand\null
2887 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2888 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
2889 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
2890 \noexpand\else
2891 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2892 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2893 \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
2894 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2895 \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
2896 \noexpand\fi
2897 \noexpand\else
2898 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2899 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2900 \@backslashchar\@backslashchar}%
2901 \noexpand\ifx\noexpand\null##3\noexpand\null
2902 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2903 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2904 \@backslashchar\noexpand\null}%
2905 \noexpand\else
2906 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2907 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2908 ##3\noexpand\null}%
2909 \noexpand\fi
2910 \noexpand\fi
2911 \noexpand\@gls@xdycheckbackslash
2912 }%
2913 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

2914 \def@gls@xdycheckbackslash

```

\@glslink If \hyperlink is not defined \@glslink ignores its first argument and just does the second argument, otherwise it is equivalent to \hyperlink.

```

2915 \ifcsundef{hyperlink}%

```

```

2916 {%
2917   \gdef\@glslink#1#2{#2}%
2918 }%
2919 {%
2920   \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
2921 }

```

`\@glstarget` If `\hypertarget` is not defined, `\@glstarget` ignores its first argument and just does the second argument, otherwise it is equivalent to `\hypertarget`.

```

2922 \newlength\gls@tmplen \ifcsundef{hypertarget}%
2923 {%
2924   \gdef\@glstarget#1#2{#2}%
2925 }%
2926 {%
2927   \gdef\@glstarget#1#2{%
2928     \settoheight{\gls@tmplen}{#2}%
2929     \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
2930   }%
2931 }

```

Glossary hyperlinks can be disabled using `\glsdisablehyper` (effect can be localised):

`\glsdisablehyper`

```

2932 \newcommand{\glsdisablehyper}{%
2933   \renewcommand*\@glslink[2]{##2}%
2934   \renewcommand*\@glstarget[2]{##2}%
2935 }

```

Glossary hyperlinks can be enabled using `\glsenablehyper` (effect can be localised):

`\glsenablehyper`

```

2936 \newcommand{\glsenablehyper}{%
2937 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
2938 \renewcommand*\@glstarget[2]{%
2939   \settoheight{\gls@tmplen}{##2}%
2940   \raisebox{\gls@tmplen}{\hypertarget{##1}{}}##2}}

```

Provide some convenience commands if not already defined:

```

2941 \providecommand{\@firstofthree}[3]{#1}
2942 \providecommand{\@secondofthree}[3]{#2}
2943 \providecommand{\@thirdofthree}[3]{#3}

```

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 if we are using the starred form:

`\gls`

```
2944 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}
```

Define the starred form:

`\@sgls`

```
2945 \newcommand*{\@sgls}[1] [] {\@gls[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

`\@gls`

```
2946 \newcommand*{\@gls}[2] [] {%
2947   \new@ifnextchar[\@gls@{#1}{#2}}{\@gls@{#1}{#2} []}%
2948 }
```

`\@gls@` Read in the final optional argument:

```
2949 \def\@gls@#1#2[#3]{%
2950   \glsdoifexists{#2}%
2951   {%
2952     \edef\@glo@type{\glsentrytype{#2}}%

2953     \let\glsifplural\@secondoftwo
2954     \let\glsifscapscase\@firstofthree
2955     \let\glsifcustomtext\@empty
2956     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`)

```
2957   \def\@glo@text{\csname gls@\@glo@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.

```
2958   \ifglsused{#2}%
2959   {%
2960     \@gls@link[#1]{#2}{\@glo@text}%
2961   }%
2962   {%
```

```

2963      \gls@checkisacronymlist\@glo@type
2964      \ifthenelse
2965      {\(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2966       \OR \NOT\boolean{glshyperfirst}
2967       }%
2968       {%
2969       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2970       }%
2971       {%
2972       \@gls@link[#1]{#2}{\@glo@text}%
2973       }%
2974       }%

```

Indicate that this entry has now been used

```

2975      \ifKV@glslink@local
2976      \glslocalunset{#2}%
2977      \else
2978      \glsunset{#2}%
2979      \fi
2980      }%
2981      }

```

`\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`

```

2982 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

2983 \newcommand*{\@sGls}[1][\@Gls[hyper=false,#1]]{

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

2984 \newcommand*{\@Gls}[2][\@Gls@{#1}{#2}]{\@Gls@{#1}{#2}[]}
2985 \new@ifnextchar{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2}[]}
2986 }

```

`\@Gls@` Read in the final optional argument:

```

2987 \def\@Gls@#1#2[#3]{%
2988   \glsdoifexists{#2}%
2989   {%
2990     \edef\@glo@type{\glstrytype{#2}}%
2991     \let\glsifplural\@secondoftwo
2992     \let\glscapscase\@secondofthree
2993     \let\glscustomtext\@empty
2994     \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in \@glo@text)

```

2995     \def\@glo@text{\csname gls@\@glo@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 hyper-
first=false package option is used.
2996     \ifglsused{#2}%
2997     {%
2998         \@gls@link[#1]{#2}{\@glo@text}%
2999     }%
3000     {%
3001         \gls@checkisacronymlist\@glo@type
3002         \ifthenelse
3003         {%
3004             \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3005             \OR \NOT\boolean{glshyperfirst}%
3006         }%
3007         {%
3008             \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3009         }%
3010         {%
3011             \@gls@link[#1]{#2}{\@glo@text}%
3012         }%
3013     }%

```

Indicate that this entry has now been used

```

3014     \ifKV@glslink@local
3015     \glslocalunset{#2}%
3016     \else
3017     \glsunset{#2}%
3018     \fi
3019 }%
3020 }

```

\GLS behaves like \gls, but the link text is converted to uppercase:

\GLS

```

3021 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

3022 \newcommand*{\@sGLS}[1] [] {\@GLS[hyper=false,#1]}

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3023 \newcommand*{\@GLS}[2] [] {%
3024     \new@ifnextchar[{\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}%
3025 }

```

\@GLS@ Read in the final optional argument:

```

3026 \def\@GLS@#1#2[#3]{%

```

```

3027 \glsdoifexists{#2}%
3028 {%
3029   \edef\@glo@type{\glsentrytype{#2}}%

3030   \let\glsifplural\@secondoftwo
3031   \let\glsifscapscase\@thirdofthree
3032   \let\glsifcustomtext\@empty
3033   \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in \@glo@text).

```

3034   \def\@glo@text{\csname gls@\@glo@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.

```

3035   \ifglsused{#2}%
3036   {%
3037     \@gls@link[#1]{#2}{\@glo@text}%
3038   }%
3039   {%
3040     \gls@checkisacronymlist\@glo@type
3041     \ifthenelse
3042     {%
3043       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3044       \OR \NOT\boolean{glshyperfirst}}{%
3045       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3046     }%
3047     {%
3048       \@gls@link[#1]{#2}{\@glo@text}%
3049     }%
3050   }%

```

Indicate that this entry has now been used

```

3051   \ifKV@glslink@local
3052   \glslocalunset{#2}%
3053   \else
3054   \glsunset{#2}%
3055   \fi
3056 }%
3057 }

```

\glspl behaves in the same way as \gls except it uses the plural form.

\glspl

```

3058 \newrobustcmd*{\glspl}{\@ifstar\@sglspl\@glspl}

```

Define the starred form:

```

3059 \newcommand*{\@sglspl}[1][\@glspl[hyper=false,#1]]{

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3060 \newcommand*{\@glspl}[2][\%
3061   \new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}%
3062 }

```

\@glspl@ Read in the final optional argument:

```

3063 \def\@glspl@#1#2[#3]{%
3064   \glsdoifexists{#2}%
3065   {%
3066     \edef\@glo@type{\glsentrytype{#2}}%

3067     \let\glsifplural\@firstoftwo
3068     \let\glsifscapscase\@firstofthree
3069     \let\glsifcustomtext\@empty
3070     \def\glsinsert{#3}%
3071 % Determine what the link text should be (this is stored in
3072 % \cs{@glo@text})
3073 % \changes{1.12}{2008 Mar 8}{now uses \cs{glsentrydescplural} and
3074 % \cs{glsentrysymbolplural} instead of \cs{glsentrydesc} and
3075 % \cs{glsentrysymbol}}
3076 % \changes{3.11a}{2013-10-15}{change to using \cs{glsentryfmt} style
3077 % commands}
3078 %   \begin{macrocode}
3079   \def\@glo@text{\csname gls@\@glo@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.

```

3080   \ifglsused{#2}%
3081   {%
3082     \@gls@link[#1]{#2}{\@glo@text}%
3083   }%
3084   {%
3085     \gls@checkisacronymlist\@glo@type
3086     \ifthenelse
3087     {%
3088       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3089       \OR \NOT\boolean{glshyperfirst}}%
3090     {%
3091       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3092     }%
3093     {%
3094       \@gls@link[#1]{#2}{\@glo@text}%
3095     }%
3096   }%
3097 }%

```

Indicate that this entry has now been used

```

3098   \ifKV@glslink@local
3099   \glslocalunset{#2}%
3100   \else

```

```

3101      \glsunset{#2}%
3102      \fi
3103    }%
3104 }

```

`\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`

```

3105 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

3106 \newcommand*{\@sGlspl}[1] [] {\@Glspl[hyper=false,#1]}

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3107 \newcommand*{\@Glspl}[2] [] {%
3108   \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}%
3109 }

```

`\@Glspl@` Read in the final optional argument:

```

3110 \def\@Glspl@#1#2[#3] {%
3111   \glsdoifexists{#2}%
3112   {%
3113     \edef\@glo@type{\glsentrytype{#2}}%
3114     \let\glsifplural\@firstoftwo
3115     \let\glsapscase\@secondofthree
3116     \let\glsustomtext\@empty
3117     \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`.

```

3118   \def\@glo@text{\csname gls@\@glo@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.

```

3119   \ifglsused{#2}%
3120   {%
3121     \@gls@link[#1]{#2}{\@glo@text}%
3122   }%
3123   {%
3124     \gls@checkisacronymlist\@glo@type
3125     \ifthenelse
3126     {%
3127       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3128       \OR \NOT\boolean{glshyperfirst}}%

```



```

3129     }%
3130     {%
3131         \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3132     }%
3133     {%
3134         \@gls@link[#1]{#2}{\@glo@text}%
3135     }%
3136 }%

```

Indicate that this entry has now been used

```

3137     \ifKV@glslink@local
3138         \glslocalunset{#2}%
3139     \else
3140         \glsunset{#2}%
3141     \fi
3142 }%
3143 }

```

`\GLSp1` behaves like `\glspl` except that all the link text is converted to uppercase.

`\GLSp1`

```

3144 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

3145 \newcommand*{\@sGLSp1}[1][\@GLSp1[hyper=false,#1]]

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3146 \newcommand*{\@GLSp1}[2][\@%
3147     \new@ifnextchar[\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}%
3148 }

```

`\@GLSp1` Read in the final optional argument:

```

3149 \def\@GLSp1@#1#2[#3]{%
3150     \glsdoifexists{#2}%
3151     {%
3152         \edef\@glo@type{\glsentrytype{#2}}%

3153         \let\glsifplural\@firstoftwo
3154         \let\glsifscapscase\@thirdofthree
3155         \let\glsifcustomtext\@empty
3156         \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in `\@glo@text`)

```

3157     \def\@glo@text{\csname gls@\@glo@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.

```

3158 \ifglsused{#2}%
3159 {%
3160 \@gls@link[#1]{#2}{\@glo@text}%
3161 }%
3162 {%
3163 \gls@checkisacronymlist\@glo@type
3164 \ifthenelse
3165 {%
3166 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3167 \OR \NOT\boolean{glshyperfirst}}%
3168 }%
3169 {%
3170 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3171 }%
3172 {%
3173 \@gls@link[#1]{#2}{\@glo@text}%
3174 }%
3175 }%

```

Indicate that this entry has now been used

```

3176 \ifKV@glslink@local
3177 \glslocalunset{#2}%
3178 \else
3179 \glsunset{#2}%
3180 \fi
3181 }%
3182 }

```

`\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:

```

3183 \newrobustcmd*{\glsdisp}{\@ifstar\sglsdisp\@glsdisp}

```

Define the starred form:

```

\@sgls
3184 \newcommand*{\sglsdisp}[1][ ]{\@glsdisp[hyper=false,#1]}

```

Defined the un-starred form.

```

\@glsdisp
3185 \newcommand*{\@glsdisp}[3][ ]{%
3186 \glsdoifexists{#2}{%
3187 \edef\@glo@type{\glsentrytype{#2}}%
3188 \let\glsifplural\@secondoftwo
3189 \let\glsifscapscase\@firstofthree
3190 \def\glscustomtext{#3}%
3191 \def\glsinsert{}}%

```

Determine what the link text should be (this is stored in \@glo@text)

```

3192 \def\@glo@text{\csname gls@\@glo@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 hyper-
first=false package option is used.
3193 \ifglsused{#2}%
3194 {%
3195 \@gls@link[#1]{#2}{\@glo@text}%
3196 }%
3197 {%
3198 \gls@checkisacronymlist\@glo@type
3199 \ifthenelse{\boolean{@glsisacronymlist}\AND
3200 \boolean{glsacrfootnote}}{\OR \NOT\boolean{glshyperfirst}}}%
3201 {%
3202 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3203 }%
3204 {%
3205 \@gls@link[#1]{#2}{\@glo@text}%
3206 }%
3207 }%

```

Indicate that this entry has now been used

```

3208 \ifKV@glslink@local
3209 \glslocalunset{#2}%
3210 \else
3211 \glsunset{#2}%
3212 \fi
3213 }%
3214 }

```

\@gls@field@link

```

3215 \newcommand{\@gls@field@link}[3]{%
3216 \glsdoifexists{#2}%
3217 {%
3218 \edef\@glo@type{\glsentrytype{#2}}%
3219 \@gls@link[#1]{#2}{#3}%
3220 }%
3221 }

```

\gls@text behaves like \gls except it always uses the value given by the text key and it doesn't mark the entry as used.

\gls@text

```

3222 \newrobustcmd*{\gls@text}{\@ifstar\@sgls@text\@gls@text}

```

Define the starred form:

```

3223 \newcommand*{\@sgls@text}[1][\@gls@text[hyper=false,#1]]{

```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3224 \newcommand*{\@glstext}[2] [] {%
3225   \new@ifnextchar[{\@glstext@{#1}{#2}}{\@glstext@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3226 \def\@glstext@#1#2[#3] {%
3227   \@gls@field@link{#1}{#2}{\glstentrytext{#2}#3}%
3228 }
```

\GLstext behaves like \glstext except the text is converted to uppercase.

\GLstext

```
3229 \newrobustcmd*{\GLstext}{\@ifstar\@sGLstext\@GLstext}
```

Define the starred form:

```
3230 \newcommand*{\@sGLstext}[1] [] {\@GLstext[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3231 \newcommand*{\@GLstext}[2] [] {%
3232   \new@ifnextchar[{\@GLstext@{#1}{#2}}{\@GLstext@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3233 \def\@GLstext@#1#2[#3] {%
3234   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentrytext{#2}#3}}%
3235 }
```

\Glstext behaves like \glstext except that the first letter of the text is converted to uppercase.

\Glstext

```
3236 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
3237 \newcommand*{\@sGlstext}[1] [] {\@Glstext[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3238 \newcommand*{\@Glstext}[2] [] {%
3239   \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3240 \def\@Glstext@#1#2[#3] {%
3241   \@gls@field@link{#1}{#2}{\Glstentrytext{#2}#3}%
3242 }
```

\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

```
3243 \newrobustcmd*{\glsfirst}{\@ifstar\@sglsfirst\@glsfirst}
```

Define the starred form:

```
3244 \newcommand*{\sglsfirst}[1] [] {\@glsfirst[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3245 \newcommand*{\@glsfirst}[2] [] {%
```

```
3246   \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3247 \def\@glsfirst@#1#2[#3] {%
```

```
3248   \@gls@field@link{#1}{#2}{\glstryfirst{#2}#3}%
```

```
3249 }
```

\Glsfirst behaves like \glsfirst except it displays the first letter in uppercase.

\Glsfirst

```
3250 \newrobustcmd*{\Glsfirst}{\@ifstar\@sGlsfirst\@Glsfirst}
```

Define the starred form:

```
3251 \newcommand*{\@sGlsfirst}[1] [] {\@Glsfirst[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3252 \newcommand*{\@Glsfirst}[2] [] {%
```

```
3253   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3254 \def\@Glsfirst@#1#2[#3] {%
```

```
3255   \@gls@field@link{#1}{#2}{\GLstryfirst{#2}#3}%
```

```
3256 }
```

\GLSfirst behaves like \Glsfirst except it displays the text in uppercase.

\GLSfirst

```
3257 \newrobustcmd*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}
```

Define the starred form:

```
3258 \newcommand*{\@sGLSfirst}[1] [] {\@GLSfirst[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3259 \newcommand*{\@GLSfirst}[2] [] {%
```

```
3260   \new@ifnextchar[{\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3261 \def\@GLSfirst@#1#2[#3] {%
```

```
3262   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstryfirst{#2}#3}}%
```

```
3263 }
```

\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`

```
3264 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
3265 \newcommand*{\@sglsplural}[1] [] {\@glsplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3266 \newcommand*{\@glsplural}[2] [] {%
```

```
3267   \new@ifnextchar[{\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3268 \def\@glsplural@#1#2[#3] {%
```

```
3269   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}%
```

```
3270 }
```

`\Glsplural` behaves like `\glsplural` except that the first letter is converted to uppercase.

`\Glsplural`

```
3271 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}
```

Define the starred form:

```
3272 \newcommand*{\@sGlsplural}[1] [] {\@Glsplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3273 \newcommand*{\@Glsplural}[2] [] {%
```

```
3274   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3275 \def\@Glsplural@#1#2[#3] {%
```

```
3276   \@gls@field@link{#1}{#2}{\Glsentryplural{#2}#3}%
```

```
3277 }
```

`\GLSplural` behaves like `\glsplural` except that the text is converted to uppercase.

`\GLSplural`

```
3278 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}
```

Define the starred form:

```
3279 \newcommand*{\@sGLSplural}[1] [] {\@GLSplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3280 \newcommand*{\@GLSplural}[2] [] {%
```

```
3281   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3282 \def\@GLSplural@#1#2[#3] {%
```

```
3283   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
```

```
3284 }
```

`\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`

```
3285 \newrobustcmd*{\glsfirstplural}{\@ifstar\sglsfirstplural\glsfirstplural}
```

Define the starred form:

```
3286 \newcommand*{\sglsfirstplural}[1][\@glsfirstplural[hyper=false,#1]]
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3287 \newcommand*{\@glsfirstplural}[2][\@%
```

```
3288 \new@ifnextchar[\@glsfirstplural@{#1}{#2}]{\@glsfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3289 \def\@glsfirstplural@#1#2[#3]{%
```

```
3290 \@gls@field@link{#1}{#2}{\glsentryfirstplural{#2}#3}%
```

```
3291 }
```

`\Glsfirstplural` behaves like `\glsfirstplural` except that the first letter is converted to uppercase.

`\Glsfirstplural`

```
3292 \newrobustcmd*{\Glsfirstplural}{\@ifstar\sglsfirstplural\Glsfirstplural}
```

Define the starred form:

```
3293 \newcommand*{\sglsfirstplural}[1][\@Glsfirstplural[hyper=false,#1]]
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3294 \newcommand*{\@Glsfirstplural}[2][\@%
```

```
3295 \new@ifnextchar[\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3296 \def\@Glsfirstplural@#1#2[#3]{%
```

```
3297 \@gls@field@link{#1}{#2}{\Glsentryfirstplural{#2}#3}%
```

```
3298 }
```

`\GLSfirstplural` behaves like `\glsfirstplural` except that the link text is converted to uppercase.

`\GLSfirstplural`

```
3299 \newrobustcmd*{\GLSfirstplural}{\@ifstar\sglsfirstplural\GLSfirstplural}
```

Define the starred form:

```
3300 \newcommand*{\sglsfirstplural}[1][\@GLSfirstplural[hyper=false,#1]]
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3301 \newcommand*{\@GLSfirstplural}[2][\@%
```

```
3302 \new@ifnextchar[\@GLSfirstplural@{#1}{#2}]{\@GLSfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3303 \def\@GLSfirstplural@#1#2[#3]{%
3304   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
3305 }
```

`\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`

```
3306 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\@glsname}
```

Define the starred form:

```
3307 \newcommand*{\@sglsname}[1] [] {\@glsname[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3308 \newcommand*{\@glsname}[2] [] {%
3309   \new@ifnextchar[{\@glsname@{#1}{#2}}{\@glsname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3310 \def\@glsname@#1#2[#3]{%
3311   \@gls@field@link{#1}{#2}{\glsentryname{#2}#3}%
3312 }
```

`\Glsname` behaves like `\glsname` except that the first letter is converted to uppercase.

`\Glsname`

```
3313 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
3314 \newcommand*{\@sGlsname}[1] [] {\@Glsname[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3315 \newcommand*{\@Glsname}[2] [] {%
3316   \new@ifnextchar[{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3317 \def\@Glsname@#1#2[#3]{%
3318   \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
3319 }
```

`\GLSname` behaves like `\glsname` except that the link text is converted to uppercase.

`\GLSname`

```
3320 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}
```

Define the starred form:

```
3321 \newcommand*{\@sGLSname}[1] [] {\@GLSname[hyper=false,#1]}
```


Defined the un-starred form. Need to determine if there is a final optional argument

```
3322 \newcommand*{\@GLSname}[2] [] {%
3323   \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3324 \def\@GLSname@#1#2[#3] {%
3325   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryname{#2}#3}}%
3326 }
```

\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

```
3327 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}
```

Define the starred form:

```
3328 \newcommand*{\@sglsdesc}[1] [] {\@glsdesc[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3329 \newcommand*{\@glsdesc}[2] [] {%
3330   \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3331 \def\@glsdesc@#1#2[#3] {%
3332   \@gls@field@link{#1}{#2}{\glsentrydesc{#2}#3}%
3333 }
```

\Glsdesc behaves like \glsdesc except that the first letter is converted to uppercase.

\Glsdesc

```
3334 \newrobustcmd*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}
```

Define the starred form:

```
3335 \newcommand*{\@sGlsdesc}[1] [] {\@Glsdesc[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3336 \newcommand*{\@Glsdesc}[2] [] {%
3337   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3338 \def\@Glsdesc@#1#2[#3] {%
3339   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
3340 }
```

\GLSdesc behaves like \glsdesc except that the link text is converted to uppercase.

\GLSdesc

```
3341 \newrobustcmd*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}
```

Define the starred form:

```
3342 \newcommand*{\@sGLSdesc}[1] [] {\@GLSdesc[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3343 \newcommand*{\@GLSdesc}[2] [] {%
```

```
3344   \new@ifnextchar[{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3345 \def\@GLSdesc@#1#2[#3] {%
```

```
3346   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydesc{#2}#3}}%
```

```
3347 }
```

`\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`

```
3348 \newrobustcmd*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}
```

Define the starred form:

```
3349 \newcommand*{\@sglsdescplural}[1] [] {\@glsdescplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3350 \newcommand*{\@glsdescplural}[2] [] {%
```

```
3351   \new@ifnextchar[{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3352 \def\@glsdescplural@#1#2[#3] {%
```

```
3353   \@gls@field@link{#1}{#2}{\glsentrydescplural{#2}#3}}%
```

```
3354 }
```

`\Glsdescplural` behaves like `\glsdescplural` except that the first letter is converted to uppercase.

`\Glsdescplural`

```
3355 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}
```

Define the starred form:

```
3356 \newcommand*{\@sGlsdescplural}[1] [] {\@Glsdescplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3357 \newcommand*{\@Glsdescplural}[2] [] {%
```

```
3358   \new@ifnextchar[{\@Glsdescplural@{#1}{#2}}{\@Glsdescplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3359 \def\@Glsdescplural@#1#2[#3] {%
```

```
3360   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}}%
```

```
3361 }
```

`\GLSdescplural` behaves like `\glsdescplural` except that the link text is converted to uppercase.

`\GLSdescplural`

```
3362 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}
```

Define the starred form:

```
3363 \newcommand*{\@sGLSdescplural}[1] [] {\@GLSdescplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3364 \newcommand*{\@GLSdescplural}[2] [] {%
```

```
3365   \new@ifnextchar[{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3366 \def\@GLSdescplural@#1#2[#3] {%
```

```
3367   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydescplural{#2}#3}}%
```

```
3368 }
```

`\glsymbol` behaves like `\gls` except it always uses the value given by the symbol key and it doesn't mark the entry as used.

`\glssymbol`

```
3369 \newrobustcmd*{\glssymbol}{\@ifstar\@sglssymbol\@glssymbol}
```

Define the starred form:

```
3370 \newcommand*{\@sglssymbol}[1] [] {\@glssymbol[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3371 \newcommand*{\@glssymbol}[2] [] {%
```

```
3372   \new@ifnextchar[{\@glssymbol@{#1}{#2}}{\@glssymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3373 \def\@glssymbol@#1#2[#3] {%
```

```
3374   \@gls@field@link{#1}{#2}{\glsentrysymbol{#2}#3}}%
```

```
3375 }
```

`\Glsymbol` behaves like `\glssymbol` except that the first letter is converted to uppercase.

`\Glsymbol`

```
3376 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
3377 \newcommand*{\@sGlsymbol}[1] [] {\@Glsymbol[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3378 \newcommand*{\@Glsymbol}[2] [] {%
```

```
3379   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3380 \def\@Glsymbol@#1#2[#3] {%
```

```
3381   \@gls@field@link{#1}{#2}{\Glsentrysymbol{#2}#3}}%
```

```
3382 }
```

`\GLSsymbol` behaves like `\glsymbol` except that the link text is converted to uppercase.

`\GLSsymbol`

```
3383 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
3384 \newcommand*{\@sGLSsymbol}[1] [] {\@GLSsymbol[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3385 \newcommand*{\@GLSsymbol}[2] [] {%
```

```
3386   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3387 \def\@GLSsymbol@#1#2[#3] {%
```

```
3388   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbol{#2}#3}}%
```

```
3389 }
```

`\glsymbolplural` behaves like `\gls` except it always uses the value given by the `symbolplural` key and it doesn't mark the entry as used.

`\glsymbolplural`

```
3390 \newrobustcmd*{\glsymbolplural}{\@ifstar\@sglsymbolplural\@glsymbolplural}
```

Define the starred form:

```
3391 \newcommand*{\@sglsymbolplural}[1] [] {\@glsymbolplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3392 \newcommand*{\@glsymbolplural}[2] [] {%
```

```
3393   \new@ifnextchar[{\@glsymbolplural@{#1}{#2}}{\@glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3394 \def\@glsymbolplural@#1#2[#3] {%
```

```
3395   \@gls@field@link{#1}{#2}{\glsentrysymbolplural{#2}#3}}%
```

```
3396 }
```

`\Glsymbolplural` behaves like `\glsymbolplural` except that the first letter is converted to uppercase.

`\Glsymbolplural`

```
3397 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
3398 \newcommand*{\@sGlsymbolplural}[1] [] {\@Glsymbolplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3399 \newcommand*{\@Glsymbolplural}[2] [] {%
```

```
3400   \new@ifnextchar[{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3401 \def\@Glsymbolplural@#1#2[#3]{%
3402   \@gls@field@link{#1}{#2}{\Glsentrysymbolplural{#2}#3}%
3403 }
```

\Glsymbolplural behaves like \glsymbolplural except that the link text is converted to uppercase.

\Glsymbolplural

```
3404 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
3405 \newcommand*{\@sGlsymbolplural}[1] [] {\@Glsymbolplural[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3406 \newcommand*{\@Glsymbolplural}[2] [] {%
3407   \new@ifnextchar[{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3408 \def\@Glsymbolplural@#1#2[#3]{%
3409   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentrysymbolplural{#2}#3}}%
3410 }
```

\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

```
3411 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}
```

Define the starred form:

```
3412 \newcommand*{\@sglsuseri}[1] [] {\@glsuseri[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3413 \newcommand*{\@glsuseri}[2] [] {%
3414   \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3415 \def\@glsuseri@#1#2[#3]{%
3416   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
3417 }
```

\Glsuseri behaves like \glsuseri except that the first letter is converted to uppercase.

\Glsuseri

```
3418 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}
```

Define the starred form:

```
3419 \newcommand*{\@sGlsuseri}[1] [] {\@Glsuseri[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3420 \newcommand*{\@Glsuseri}[2] [] {%
3421   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3422 \def\@Glsuseri@#1#2[#3] {%
3423   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
3424 }
```

\Glsuseri behaves like \glsuseri except that the link text is converted to uppercase.

\Glsuseri

```
3425 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}
```

Define the starred form:

```
3426 \newcommand*{\@sGlsuseri}[1] [] {\@Glsuseri[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3427 \newcommand*{\@Glsuseri}[2] [] {%
3428   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3429 \def\@Glsuseri@#1#2[#3] {%
3430   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseri{#2}#3}}%
3431 }
```

\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

```
3432 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}
```

Define the starred form:

```
3433 \newcommand*{\@sglsuserii}[1] [] {\@glsuserii[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3434 \newcommand*{\@glsuserii}[2] [] {%
3435   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3436 \def\@glsuserii@#1#2[#3] {%
3437   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
3438 }
```

\Glsuserii behaves like \glsuserii except that the first letter is converted to uppercase.

\Glsuserii

```
3439 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3440 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3441 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3442   \new@ifnextchar[\@Glsuserii@{#1}{#2}]{\@Glsuserii@{#1}{#2} []}}
```

Read in the final optional argument:

```
3443 \def\@Glsuserii@#1#2[#3] {%
```

```
3444   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
```

```
3445 }
```

\Glsuserii behaves like \glsuserii except that the link text is converted to uppercase.

\Glsuserii

```
3446 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3447 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3448 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3449   \new@ifnextchar[\@Glsuserii@{#1}{#2}]{\@Glsuserii@{#1}{#2} []}}
```

Read in the final optional argument:

```
3450 \def\@Glsuserii@#1#2[#3] {%
```

```
3451   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3452 }
```

\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

```
3453 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}
```

Define the starred form:

```
3454 \newcommand*{\@sglsuseriii}[1] [] {\@glsuseriii[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3455 \newcommand*{\@glsuseriii}[2] [] {%
```

```
3456   \new@ifnextchar[\@glsuseriii@{#1}{#2}]{\@glsuseriii@{#1}{#2} []}}
```

Read in the final optional argument:

```
3457 \def\@glsuseriii@#1#2[#3] {%
```

```
3458   \@gls@field@link{#1}{#2}{\glsentryuseriii{#2}#3}%
```

```
3459 }
```

\Glsuseriii behaves like \glsuseriii except that the first letter is converted to uppercase.

`\Glsuseriii`

```
3460 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
3461 \newcommand*{\@sGlsuseriii}[1] [] {\@Glsuseriii[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3462 \newcommand*{\@Glsuseriii}[2] [] {%
```

```
3463   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3464 \def\@Glsuseriii@#1#2[#3]{%
```

```
3465   \@gls@field@link{#1}{#2}{\Glsentryuseriii{#2}#3}%
```

```
3466 }
```

`\GLSuseriii` behaves like `\glsuseriii` except that the link text is converted to uppercase.

`\GLSuseriii`

```
3467 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}
```

Define the starred form:

```
3468 \newcommand*{\@sGLSuseriii}[1] [] {\@GLSuseriii[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3469 \newcommand*{\@GLSuseriii}[2] [] {%
```

```
3470   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3471 \def\@GLSuseriii@#1#2[#3]{%
```

```
3472   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseriii{#2}#3}}%
```

```
3473 }
```

`\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`

```
3474 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
3475 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3476 \newcommand*{\@glsuseriv}[2] [] {%
```

```
3477   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3478 \def\@glsuseriv@#1#2[#3]{%
```

```
3479   \@gls@field@link{#1}{#2}{\glsentryuseriv{#2}#3}%
```

```
3480 }
```


`\Glsuseriv` behaves like `\glsuseriv` except that the first letter is converted to uppercase.

`\Glsuseriv`

```
3481 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
3482 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3483 \newcommand*{\@Glsuseriv}[2] [] {%
```

```
3484   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3485 \def\@Glsuseriv@#1#2[#3] {%
```

```
3486   \@gls@field@link{#1}{#2}{\Glsentryuseriv{#2}#3}%
```

```
3487 }
```

`\GLSuseriv` behaves like `\glsuseriv` except that the link text is converted to uppercase.

`\GLSuseriv`

```
3488 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}
```

Define the starred form:

```
3489 \newcommand*{\@sGLSuseriv}[1] [] {\@GLSuseriv[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3490 \newcommand*{\@GLSuseriv}[2] [] {%
```

```
3491   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3492 \def\@GLSuseriv@#1#2[#3] {%
```

```
3493   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseriv{#2}#3}}%
```

```
3494 }
```

`\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`

```
3495 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}
```

Define the starred form:

```
3496 \newcommand*{\@sglsuserv}[1] [] {\@glsuserv[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3497 \newcommand*{\@glsuserv}[2] [] {%
```

```
3498   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3499 \def\@glsuserv@#1#2[#3]{%
3500   \@gls@field@link{#1}{#2}{\glsentryuserv{#2}#3}%
3501 }
```

\Glsuserv behaves like \glsuserv except that the first letter is converted to uppercase.

\Glsuserv

```
3502 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
3503 \newcommand*{\@sGlsuserv}[1][\@Glsuserv[hyper=false,#1]]
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3504 \newcommand*{\@Glsuserv}[2][\@%
3505 \new@ifnextchar[\@Glsuserv@{#1}{#2}]{\@Glsuserv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3506 \def\@Glsuserv@#1#2[#3]{%
3507   \@gls@field@link{#1}{#2}{\glsentryuserv{#2}#3}%
3508 }
```

\GLSuserv behaves like \glsuserv except that the link text is converted to uppercase.

\GLSuserv

```
3509 \newrobustcmd*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}
```

Define the starred form:

```
3510 \newcommand*{\@sGLSuserv}[1][\@GLSuserv[hyper=false,#1]]
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3511 \newcommand*{\@GLSuserv}[2][\@%
3512 \new@ifnextchar[\@GLSuserv@{#1}{#2}]{\@GLSuserv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3513 \def\@GLSuserv@#1#2[#3]{%
3514   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserv{#2}#3}}%
3515 }
```

\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

```
3516 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
3517 \newcommand*{\@sglsuservi}[1][\@glsuservi[hyper=false,#1]]
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3518 \newcommand*{\@glsuservi}[2] [] {%
3519   \new@ifnextchar[{\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3520 \def\@glsuservi@#1#2[#3] {%
3521   \@gls@field@link{#1}{#2}{\glsentryuservi{#2}#3}%
3522 }
```

\Glsuservi behaves like \glsuservi except that the first letter is converted to uppercase.

\Glsuservi

```
3523 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}
```

Define the starred form:

```
3524 \newcommand*{\@sGlsuservi}[1] [] {\@Glsuservi[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3525 \newcommand*{\@Glsuservi}[2] [] {%
3526   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3527 \def\@Glsuservi@#1#2[#3] {%
3528   \@gls@field@link{#1}{#2}{\Glsentryuservi{#2}#3}%
3529 }
```

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

```
3530 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}
```

Define the starred form:

```
3531 \newcommand*{\@sGLSuservi}[1] [] {\@GLSuservi[hyper=false,#1]}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3532 \newcommand*{\@GLSuservi}[2] [] {%
3533   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3534 \def\@GLSuservi@#1#2[#3] {%
3535   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuservi{#2}#3}}%
3536 }
```

Now deal with acronym related keys. First the short form:

\acrshort

```
3537 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\ns@acrshort}
```

Define the starred form:

```
3538 \newcommand*{\s@acrshort}[2] [] {%
3539   \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}{%
3540     {\@acrshort{hyper=false,#1}{#2} []}%
3541 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3542 \newcommand*{\ns@acrshort}[2] [] {%
3543   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}%
3544 }
```

Read in the final optional argument:

```
3545 \def\@acrshort#1#2[#3] {%
3546   \glsdoifexists{#2}%
3547   {%
3548     \edef\@glo@type{\glsentrytype{#2}}%

3549     \let\glsifplural\@secondoftwo
3550     \let\glsupcase\@firstofthree
3551     \let\glsinsert\@empty
3552     \def\glscustomtext{%
3553       \acronymfont{\glsentryshort{#2}}#3%
3554     }%
```

Call \@gls@link

```
3555   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3556   }%
3557 }
```

\Acrshort

```
3558 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}
```

Define the starred form:

```
3559 \newcommand*{\s@Acrshort}[2] [] {%
3560   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}{%
3561     {\@Acrshort{hyper=false,#1}{#2} []}%
3562 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3563 \newcommand*{\ns@Acrshort}[2] [] {%
3564   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
3565 }
```

Read in the final optional argument:

```
3566 \def\@Acrshort#1#2[#3] {%
3567   \glsdoifexists{#2}%
3568   {%
3569     \edef\@glo@type{\glsentrytype{#2}}%
```

```

3570 \def\glslabel{#2}%
3571 \let\glsifplural\@secondoftwo
3572 \let\glscapscase\@secondofthree
3573 \let\glsinsert\@empty
3574 \def\glscustomtext{%
3575 \acronymfont{\Glsentryshort{#2}}#3%
3576 }%

Call \@gls@link
3577 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3578 }%
3579 }

```

\ACRshort

```

3580 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\@ns@ACRshort}

```

Define the starred form:

```

3581 \newcommand*{\s@ACRshort}[2][]{%
3582 \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%
3583 {\@ACRshort{hyper=false,#1}{#2}[]}%
3584 }

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3585 \newcommand*{\ns@ACRshort}[2][]{%
3586 \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2}[]}%
3587 }

```

Read in the final optional argument:

```

3588 \def\@ACRshort#1#2[#3]{%
3589 \glsdoifexists{#2}%
3590 {%
3591 \edef\@glo@type{\glsentrytype{#2}}%

3592 \def\glslabel{#2}%
3593 \let\glsifplural\@secondoftwo
3594 \let\glscapscase\@thirdofthree
3595 \let\glsinsert\@empty
3596 \def\glscustomtext{%
3597 \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3598 }%

```

Call \@gls@link

```

3599 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3600 }%
3601 }

```

Short plural:

\acrshortpl

```

3602 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\@ns@acrshortpl}

```

Define the starred form:

```
3603 \newcommand*{\s@acrshortpl}[2] [] {%
3604   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}}%
3605   {\@acrshortpl{hyper=false,#1}{#2} []}%
3606 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3607 \newcommand*{\ns@acrshortpl}[2] [] {%
3608   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
3609 }
```

Read in the final optional argument:

```
3610 \def\@acrshortpl#1#2[#3] {%
3611   \glsdoifexists{#2}%
3612   {%
3613     \edef\@glo@type{\glsentrytype{#2}}%

3614     \def\glslabel{#2}%
3615     \let\glsifplural\@firstoftwo
3616     \let\glsupcase\@firstofthree
3617     \let\glsinsert\@empty
3618     \def\glscustomtext{%
3619       \acronymfont{\glsentryshortpl{#2}}#3%
3620     }%
```

Call \@gls@link

```
3621   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3622   }%
3623 }
```

\Acrshortpl

```
3624 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\ns@Acrshortpl}
```

Define the starred form:

```
3625 \newcommand*{\s@Acrshortpl}[2] [] {%
3626   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}}%
3627   {\@Acrshortpl{hyper=false,#1}{#2} []}%
3628 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3629 \newcommand*{\ns@Acrshortpl}[2] [] {%
3630   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
3631 }
```

Read in the final optional argument:

```
3632 \def\@Acrshortpl#1#2[#3] {%
3633   \glsdoifexists{#2}%
3634   {%
3635     \edef\@glo@type{\glsentrytype{#2}}%
```

```

3636 \def\glslabel{#2}%
3637 \let\glsifplural\@firstoftwo
3638 \let\glscapscase\@secondofthree
3639 \let\glsinsert\@empty
3640 \def\glscustomtext{%
3641 \acronymfont{\Glsentryshortpl{#2}}#3%
3642 }%

Call \@gls@link
3643 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3644 }%
3645 }

```

\ACRshortpl

```

3646 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}

```

Define the starred form:

```

3647 \newcommand*{\s@ACRshortpl}[2][]{%
3648 \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}%
3649 {\@ACRshortpl{hyper=false,#1}{#2}[]}%
3650 }

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3651 \newcommand*{\ns@ACRshortpl}[2][]{%
3652 \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2}[]}%
3653 }

```

Read in the final optional argument:

```

3654 \def\@ACRshortpl#1#2[#3]{%
3655 \glsdoifexists{#2}%
3656 {%
3657 \edef\@glo@type{\glsentrytype{#2}}%

3658 \def\glslabel{#2}%
3659 \let\glsifplural\@firstoftwo
3660 \let\glscapscase\@thirdofthree
3661 \let\glsinsert\@empty
3662 \def\glscustomtext{%
3663 \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}%
3664 }%

```

Call \@gls@link

```

3665 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3666 }%
3667 }

```

\acrlong

```

3668 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}

```

Define the starred form:

```
3669 \newcommand*{\s@acrlong}[2] [] {%
3670   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}}%
3671   {\@acrlong{hyper=false,#1}{#2} []}%
3672 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3673 \newcommand*{\ns@acrlong}[2] [] {%
3674   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3675 }
```

Read in the final optional argument:

```
3676 \def\@acrlong#1#2[#3] {%
3677   \glsdoifexists{#2}%
3678   {%
3679     \edef\@glo@type{\glsentrytype{#2}}%

3680     \def\glslabel{#2}%
3681     \let\glsifplural\@secondoftwo
3682     \let\glscapscase\@firstofthree
3683     \let\glsinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```
3684   \def\glscustomtext{%
3685     \glsentrylong{#2}#3%
3686   }%
```

Call \@gls@link

```
3687   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3688   }%
3689 }
```

\Acrlong

```
3690 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\ns@Acrlong}
```

Define the starred form:

```
3691 \newcommand*{\s@Acrlong}[2] [] {%
3692   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}}%
3693   {\@Acrlong{hyper=false,#1}{#2} []}%
3694 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3695 \newcommand*{\ns@Acrlong}[2] [] {%
3696   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3697 }
```


Read in the final optional argument:

```

3698 \def\@Acrlong#1#2[#3]{%
3699   \glsdoifexists{#2}%
3700   {%
3701     \edef\@glo@type{\glsentrytype{#2}}%

3702     \def\glslabel{#2}%
3703     \let\glsifplural\@secondoftwo
3704     \let\glscapscase\@secondofthree
3705     \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3706     \def\glscustomtext{%
3707       \Glsentrylong{#2}#3%
3708     }%

```

Call \@gls@link

```

3709     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3710   }%
3711 }

```

\ACRlong

```

3712 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\@ns@ACRlong}

```

Define the starred form:

```

3713 \newcommand*{\s@ACRlong}[2][]{%
3714   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}{%
3715     {\@ACRlong{hyper=false,#1}{#2} []}%
3716 }

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3717 \newcommand*{\ns@ACRlong}[2][]{%
3718   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
3719 }

```

Read in the final optional argument:

```

3720 \def\@ACRlong#1#2[#3]{%
3721   \glsdoifexists{#2}%
3722   {%
3723     \edef\@glo@type{\glsentrytype{#2}}%

3724     \def\glslabel{#2}%
3725     \let\glsifplural\@secondoftwo
3726     \let\glscapscase\@thirdofthree
3727     \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3728 \def\glscustomtext{%
3729 \mfirstucMakeUppercase{\glsentrylong{#2}#3}%
3730 }%
    Call \@gls@link
3731 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3732 }%
3733 }

```

Short plural:

\acrlongpl

```

3734 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}

```

Define the starred form:

```

3735 \newcommand*{\s@acrlongpl}[2] [] {%
3736 \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
3737 {\@acrlongpl{hyper=false,#1}{#2} []}%
3738 }

```

Defined the un-starred form. Need to determine if there is a final optional argument

```

3739 \newcommand*{\ns@acrlongpl}[2] [] {%
3740 \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
3741 }

```

Read in the final optional argument:

```

3742 \def\@acrlongpl#1#2[#3] {%
3743 \glsoifexists{#2}%
3744 {%
3745 \edef\@glo@type{\glsentrytype{#2}}%
3746 \def\glslabel{#2}%
3747 \let\glsifplural\@firstoftwo
3748 \let\glscapscase\@firstofthree
3749 \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3750 \def\glscustomtext{%
3751 \glsentrylongpl{#2}#3%
3752 }%

```

Call \@gls@link

```

3753 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3754 }%
3755 }

```

\Acrlongpl

```

3756 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}

```

Define the starred form:

```
3757 \newcommand*{\s@Acrlongpl}[2] [] {%
3758   \new@ifnextchar[{\@Acrlongpl{hyper=false,#1}{#2}}{%
3759     {\@Acrlongpl{hyper=false,#1}{#2} []}%
3760 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3761 \newcommand*{\ns@Acrlongpl}[2] [] {%
3762   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
3763 }
```

Read in the final optional argument:

```
3764 \def\@Acrlongpl#1#2[#3] {%
3765   \glsdoifexists{#2}%
3766   {%
3767     \edef\@glo@type{\glsentrytype{#2}}%

3768     \def\glslabel{#2}%
3769     \let\glsifplural\@firstoftwo
3770     \let\glscapscase\@secondofthree
3771     \let\glsinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```
3772   \def\glscustomtext{%
3773     \Glsentrylongpl{#2}#3%
3774   }%
```

Call \@gls@link

```
3775   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3776   }%
3777 }
```

\ACRlongpl

```
3778 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}
```

Define the starred form:

```
3779 \newcommand*{\s@ACRlongpl}[2] [] {%
3780   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}{%
3781     {\@ACRlongpl{hyper=false,#1}{#2} []}%
3782 }
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3783 \newcommand*{\ns@ACRlongpl}[2] [] {%
3784   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3785 }
```

Read in the final optional argument:

```

3786 \def\@ACRlongpl#1#2[#3]{%
3787   \glsdoifexists{#2}%
3788   {%
3789     \edef\@glo@type{\glentrytype{#2}}%

3790   \def\glslabel{#2}%
3791   \let\glsifplural\@firstoftwo
3792   \let\glscapscase\@thirdofthree
3793   \let\glsinsert\@empty

```

Bug fix v4.02 removed `\acronymfont` from `\glscustomtext` (`\acronymfont` only designed for short form).

```

3794   \def\glscustomtext{%
3795     \mfirstucMakeUppercase{\glentrylongpl{#2}#3}%
3796   }%

```

Call `\@gls@link`

```

3797   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3798   }%
3799 }

```

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>}
```

```

3800 \newcommand*{\@gls@entry@field}[2]{%
3801   \csname glo@\glsdetoklabel{#1}@#2\endcsname
3802 }

```

`\@Gls@entry@field` Generic first letter uppercase version.

```
\@Gls@entry@field{<label>}{<field>}
```

```

3803 \newcommand*{\@Gls@entry@field}[2]{%
3804   \letcs\@glo@text{glo@\glsdetoklabel{#1}@#2}%
3805   \xmakefirstuc{\@glo@text}%
3806 }

```

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
3807 \newcommand*{\glsentryname}[1]{\@gls@entry@field{#1}{name}}
```

```
\Glsentryname
3808 \newrobustcmd*{\Glsentryname}[1]{%
3809   \@Gls@entry@field{#1}{name}%
3810 }
```

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 unless you used `description=false` in the `sanitize` package option you may get unexpected results if the description key contained any commands.

```
\glsentrydesc
3811 \newcommand*{\glsentrydesc}[1]{\@gls@entry@field{#1}{desc}}
```

```
\Glsentrydesc
3812 \newrobustcmd*{\Glsentrydesc}[1]{%
3813   \@Gls@entry@field{#1}{desc}%
3814 }
```

Plural form:

```
\glsentrydescplural
3815 \newcommand*{\glsentrydescplural}[1]{%
3816   \@gls@entry@field{#1}{descplural}%
3817 }
```

```
\Glsentrydescplural
3818 \newrobustcmd*{\Glsentrydescplural}[1]{%
3819   \@Gls@entry@field{#1}{descplural}%
3820 }
```

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
3821 \newcommand*{\glsentrytext}[1]{\@gls@entry@field{#1}{text}}
```

```
\Glsentrytext
3822 \newrobustcmd*{\Glsentrytext}[1]{%
3823   \@Gls@entry@field{#1}{text}%
3824 }
```

Get the plural form:

```
\glsentryplural
3825 \newcommand*{\glsentryplural}[1]{%
3826   \@gls@entry@field{#1}{plural}%
3827 }
```

`\Glsentryplural`

```
3828 \newrobustcmd*{\Glsentryplural}[1]{%
3829   \@Gls@entry@field{#1}{plural}%
3830 }
```

Get the symbol associated with this entry. The argument is the label associated with the entry.

`\glsentrysymbol`

```
3831 \newcommand*{\glsentrysymbol}[1]{%
3832   \@Gls@entry@field{#1}{symbol}%
3833 }
```

`\Glsentrysymbol`

```
3834 \newrobustcmd*{\Glsentrysymbol}[1]{%
3835   \@Gls@entry@field{#1}{symbol}%
3836 }
```

Plural form:

`\glsentrysymbolplural`

```
3837 \newcommand*{\glsentrysymbolplural}[1]{%
3838   \@Gls@entry@field{#1}{symbolplural}%
3839 }
```

`\Glsentrysymbolplural`

```
3840 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3841   \@Gls@entry@field{#1}{symbolplural}%
3842 }
```

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`

```
3843 \newcommand*{\glsentryfirst}[1]{%
3844   \@Gls@entry@field{#1}{first}%
3845 }
```

`\Glsentryfirst`

```
3846 \newrobustcmd*{\Glsentryfirst}[1]{%
3847   \@Gls@entry@field{#1}{first}%
3848 }
```

Get the plural form (as specified by the firstplural key when the entry was defined).

`\glsentryfirstplural`

```
3849 \newcommand*{\glsentryfirstplural}[1]{%
3850   \@Gls@entry@field{#1}{firstpl}%
3851 }
```

Glsentryfirstplural

```
3852 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3853   \@Gls@entry@field{#1}{firstpl}%
3854 }
```

Display the glossary type with which this entry is associated (as specified by the type key used when the entry was defined)

\glsentrytype

```
3855 \newcommand*{\glsentrytype}[1]{\@Gls@entry@field{#1}{type}}
```

Display the sort text used for this entry. Note that the sort key is sanitized, so unexpected results may occur if the sort key contained commands.

\glsentrysort

```
3856 \newcommand*{\glsentrysort}[1]{%
3857   \@Gls@entry@field{#1}{sort}%
3858 }
```

\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.

```
3859 \newcommand*{\glsentryuseri}[1]{%
3860   \@Gls@entry@field{#1}{useri}%
3861 }
```

\Glsentryuseri

```
3862 \newrobustcmd*{\Glsentryuseri}[1]{%
3863   \@Gls@entry@field{#1}{useri}%
3864 }
```

\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.

```
3865 \newcommand*{\glsentryuserii}[1]{%
3866   \@Gls@entry@field{#1}{userii}%
3867 }
```

\Glsentryuserii

```
3868 \newrobustcmd*{\Glsentryuserii}[1]{%
3869   \@Gls@entry@field{#1}{userii}%
3870 }
```

\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.

```
3871 \newcommand*{\glsentryuseriii}[1]{%
3872   \@Gls@entry@field{#1}{useriii}%
3873 }
```

\Glsentryuseriii

```
3874 \newrobustcmd*{\Glsentryuseriii}[1]{%
3875   \@Gls@entry@field{#1}{useriii}%
3876 }
```

\glsentryuseriv Get the fourth user key (as specified by the user4 when the entry was defined).
The argument is the label associated with the entry.

```
3877 \newcommand*{\glsentryuseriv}[1]{%
3878   \@Gls@entry@field{#1}{useriv}%
3879 }
```

\Glsentryuseriv

```
3880 \newrobustcmd*{\Glsentryuseriv}[1]{%
3881   \@Gls@entry@field{#1}{useriv}%
3882 }
```

\glsentryuserv Get the fifth user key (as specified by the user5 when the entry was defined).
The argument is the label associated with the entry.

```
3883 \newcommand*{\glsentryuserv}[1]{%
3884   \@Gls@entry@field{#1}{userv}%
3885 }
```

\Glsentryuserv

```
3886 \newrobustcmd*{\Glsentryuserv}[1]{%
3887   \@Gls@entry@field{#1}{userv}%
3888 }
```

\glsentryuservi Get the sixth user key (as specified by the user6 when the entry was defined).
The argument is the label associated with the entry.

```
3889 \newcommand*{\glsentryuservi}[1]{%
3890   \@Gls@entry@field{#1}{uservi}%
3891 }
```

\Glsentryuservi

```
3892 \newrobustcmd*{\Glsentryuservi}[1]{%
3893   \@Gls@entry@field{#1}{uservi}%
3894 }
```

\glsentryshort Get the short key (as specified by the short the entry was defined). The argument is the label associated with the entry.

```
3895 \newcommand*{\glsentryshort}[1]{\@Gls@entry@field{#1}{short}}
```

\Glsentryshort

```
3896 \newrobustcmd*{\Glsentryshort}[1]{%
3897   \@Gls@entry@field{#1}{short}%
3898 }
```


`\glsentryshortpl` Get the short plural key (as specified by the shortplural the entry was defined).
The argument is the label associated with the entry.

```
3899 \newcommand*{\glsentryshortpl}[1]{\@gls@entry@field{#1}{shortpl}}
```

`\Glsentryshortpl`

```
3900 \newrobustcmd*{\Glsentryshortpl}[1]{%
3901   \@Gls@entry@field{#1}{shortpl}}%
3902 }
```

`\glsentrylong` Get the long key (as specified by the long the entry was defined). The argument is the label associated with the entry.

```
3903 \newcommand*{\glsentrylong}[1]{\@gls@entry@field{#1}{long}}
```

`\Glsentrylong`

```
3904 \newrobustcmd*{\Glsentrylong}[1]{%
3905   \@Gls@entry@field{#1}{long}}%
3906 }
```

`\glsentrylongpl` Get the long plural key (as specified by the longplural the entry was defined).
The argument is the label associated with the entry.

```
3907 \newcommand*{\glsentrylongpl}[1]{\@gls@entry@field{#1}{longpl}}
```

`\Glsentrylongpl`

```
3908 \newrobustcmd*{\Glsentrylongpl}[1]{%
3909   \@Gls@entry@field{#1}{longpl}}%
3910 }
```

Short cut macros to access full form:

`\glsentryfull`

```
3911 \newcommand*{\glsentryfull}[1]{%
3912   \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3913 }
```

`\Glsentryfull`

```
3914 \newrobustcmd*{\Glsentryfull}[1]{%
3915   \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3916 }
```

`\glsentryfullpl`

```
3917 \newcommand*{\glsentryfullpl}[1]{%
3918   \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3919 }
```

`\Glsentryfullpl`

```
3920 \newrobustcmd*{\Glsentryfullpl}[1]{%
3921   \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3922 }
```

<code>\glstentrynumberlist</code>	Displays the number list as is.
-----------------------------------	---------------------------------

`\displaynumberlist` Formats the number list for the given entry label. Doesn't work with hyperref.

```

3941 \newcommand*{\glsdisplaynumberlist}[1]{%
3942   \glsdoifexists{#1}%
3943   {%
3944     \bgroup
3945       \edef\@glo@label{\glsdetoklabel{#1}}%
3946       \let\@orglsnumberformat\glsnumberformat
3947       \def\glsnumberformat##1{##1}%
3948       \protected@edef\the@numberlist{%
3949         \csname glo@\@glo@label @numberlist\endcsname}%
3950       \def\@gls@numlist@sep{}%
3951       \def\@gls@numlist@nextsep{}%
3952       \def\@gls@numlist@lastsep{}%
3953       \def\@gls@thislist{}%
3954       \def\@gls@donext@def{}%
3955       \renewcommand\do[1]{%
3956         \protected@edef\@gls@thislist{%
3957           \@gls@thislist
3958           \noexpand\@gls@numlist@sep
3959           ##1%
3960         }%
3961         \let\@gls@numlist@sep\@gls@numlist@nextsep
3962         \def\@gls@numlist@nextsep{\glsnumlistsep}%
3963         \@gls@donext@def
3964         \def\@gls@donext@def{%
3965           \def\@gls@numlist@lastsep{\glsnumlistlastsep}%
3966         }%
3967       }%

```

```

3968      \expandafter \glsnumlistparser \expandafter{\the@numberlist}%
3969      \let\@gls@numlist@sep\@gls@numlist@lastsep
3970      \@gls@thislist
3971      \egroup
3972    }%
3973  }
3974 }

```

`\glsnumlistsep`

```
3975 \newcommand*{\glsnumlistsep}{, }
```

`\glsnumlistlastsep`

```
3976 \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.

```

3977 \newcommand*{\glshyperlink}[2][\glsentrytext{\@glo@label}]{%
3978   \def\@glo@label{#2}%
3979   \@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`:

```
3980 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
3981 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
3982 \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`

```

3983 \newrobustcmd*{\glsadd}[2][ ]{%
3984   \glsdoifexists{#2}%
3985   {%
3986     \def\@glsnumberformat{\glsnumberformat}%
3987     \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
3988     \setkeys{glossadd}{#1}%

```

Store the entry's counter in \theglentrycounter

```
3989 \gls@saveentrycounter
3990 \do@wrglossary{#2}%
3991 }%
3992 }
```

\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

```
3993 \newrobustcmd*{\glsaddall}[1] [] {%
3994 \edef\@glo@type{\@glo@types}%
3995 \setkeys{glossadd}{#1}%
3996 \forallglsentries[\@glo@type]{\@glo@entry}{%
3997 \glsadd[#1]{\@glo@entry}%
3998 }%
3999 }
```

\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.

```
4000 \newrobustcmd*{\glsaddallunused}[1] [\@glo@types] {%
4001 \forallglsentries[#1]{\@glo@entry}%
4002 {%
4003 \ifglsused{\@glo@entry}{\@glsadd[format=@gobble]{\@glo@entry}}%
4004 }%
4005 }
```

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.

```
4006 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```
4007 \edef\glsclosebrace{\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.

```
4008 \edef\glsquote#1{\string"#1\string"}
```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for xindy.

```
4009 \ifglxindy
4010   \newcommand*{\@glsfirstletter}{A}
4011 \fi
```

`\stLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```
4012 \ifglxindy
4013   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4014     \renewcommand*{\@glsfirstletter}{#1}}
4015 \else
4016   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4017     \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
4018 \fi
```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```
4019 \newcommand*{\@glsminrange}{2}
```

`\setXdyMinRangeLength` 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.

```
4020 \ifglxindy
4021   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4022     \renewcommand*{\@glsminrange}{#1}}
4023 \else
4024   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4025     \glsnoxywarning\GlsSetXdyMinRangeLength}
4026 \fi
```

`\writeist`

```
4027 \ifglxindy
```

Code to use if xindy is required.

```

4028 \def\writeist{%
Update attributes list
4029 \@gls@addpredefinedattributes
Open the file.
4030 \openout\glswrite=\istfilename
Write header comment at the start of the file
4031 \write\glswrite{;; xindy style file created by the glossaries
4032 package}%
4033 \write\glswrite{;; for document '\jobname' on
4034 \the\year-\the\month-\the\day}%
Specify the required styles
4035 \write\glswrite{^^J; required styles^^J}
4036 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4037 \ifx\@xdystyle\@empty
4038 \else
4039 \protected@write\glswrite{{(require
4040 \string"\@xdystyle.xdy\string")}}%
4041 \fi
4042 }%
List the allowed attributes (possible values used by the format key)
4043 \write\glswrite{^^J%
4044 ; list of allowed attributes (number formats)^^J}%
4045 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
Define any additional alphabets
4046 \write\glswrite{^^J; user defined alphabets^^J}%
4047 \write\glswrite{\@xdyuseralphabets}%
Define location classes.
4048 \write\glswrite{^^J; location class definitions^^J}%
As from version 3.0, locations are now specified as {\<Hprefix>}{\<number>}, so
need to add all possible combinations of location types.
4049 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
Case were \<Hprefix> is empty:
4050 \protected@write\glswrite{{(define-location-class
4051 \string"\@gls@classI\string"^^J\space\space\space
4052 (
4053 :sep "{}{"
4054 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4055 :sep "}"
4056 )
4057 ^^J\space\space\space
4058 :min-range-length \@glsminrange^^J%
4059 )
4060 }%

```

Nested iteration over all classes:

```

4061     {%
4062         \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4063             \protected@write\glswrite{}\{(define-location-class
4064                 \string"\@gls@classII-\@gls@classI\string"
4065                 ^^J\space\space\space
4066                 (
4067                     :sep "{"
4068                     \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
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         }%
4077     }%
4078 }%
4079 }%

```

User defined location classes (needs checking for new location format).

```

4080 \write\glswrite{^^J; user defined location classes}%
4081 \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 `\glsseeformat` which xindy won't recognise.)

```

4082 \write\glswrite{^^J; define cross-reference class^^J}%
4083 \write\glswrite{(define-crossref-class \string"see\string"
4084                 :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.)

```

4085 \write\glswrite{(markup-crossref-list
4086                 :class \string"see\string"^^J\space\space\space
4087                 :open \string"\string\glsseeformat\string"
4088                 :close \string"{}\string")}%

```

List the order to sort the classes.

```

4089 \write\glswrite{^^J; define the order of the location classes}%
4090 \write\glswrite{(define-location-class-order
4091                 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4092 \write\glswrite{^^J; define the glossary markup^^J}%

4093 \write\glswrite{(markup-index^^J\space\space\space
4094                 :open \string"\string

```

```

4095      \glossarysection[\string\glossarytoctitle]{\string
4096      \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)

```

4097      \@for\@this@ctr:=\@xdycounters\do{%
4098      {%
4099      \@for\@this@attr:=\@xdyattributelist\do{%
4100      \protected@write\glswrite{}\{\string\providecommand*%
4101      \expandafter\string
4102      \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4103      {%
4104      \string\setentrycounter
4105      [\expandafter\@gobble\string\#1]{\@this@ctr}%
4106      \expandafter\string
4107      \csname\@this@attr\endcsname
4108      {\expandafter\@gobble\string\#2}%
4109      }%
4110      }%
4111      }%
4112      }%
4113      }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

4114      \write\glswrite{%
4115      \string\begin
4116      {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4117      \space\space:close \string"\expandafter\@gobble
4118      \string\%\string~n\string
4119      \end{theglossary}\string\glossarypostamble
4120      \string~n\string" ^^J\space\space\space
4121      :tree)}}%

```

Specify what to put between letter groups

```

4122      \write\glswrite{(markup-letter-group-list
4123      :sep \string"\string\glsgroupskip\string~n\string"))}%

```

Specify what to put between entries

```

4124      \write\glswrite{(markup-indexentry
4125      :open \string"\string\relax \string\glsresetentrylist
4126      \string~n\string"))}%

```

Specify how to format entries

```

4127      \write\glswrite{(markup-locclass-list :open
4128      \string"\glsopenbrace\string\glossaryentrynumbers
4129      \glsopenbrace\string\relax\space \string"^^J\space\space\space
4130      :sep \string", \string"
4131      :close \string"\glsclosebrace\glsresetentrylist\string"))}%

```

Specify how to separate location numbers

```

4132      \write\glswrite{(markup-locref-list
4133      :sep \string"\string\delimN\space\string"))}%

```


Specify how to indicate location ranges

```
4134 \write\glswrite{(markup-range
4135 :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.

```
4136 \@onelevel@sanitize\gls@suffixF
4137 \@onelevel@sanitize\gls@suffixFF
4138 \ifx\gls@suffixF\@empty
4139 \else
4140 \write\glswrite{(markup-range
4141 :close "\gls@suffixF" :length 1 :ignore-end)}}%
4142 \fi
4143 \ifx\gls@suffixFF\@empty
4144 \else
4145 \write\glswrite{(markup-range
4146 :close "\gls@suffixFF" :length 2 :ignore-end)}}%
4147 \fi
```

Specify how to format locations.

```
4148 \write\glswrite{^^J; define format to use for locations^^J}%
4149 \write\glswrite{\@xdylocref}}%
```

Specify how to separate letter groups.

```
4150 \write\glswrite{^^J; define letter group list format^^J}%
4151 \write\glswrite{(markup-letter-group-list
4152 :sep \string\string\glsgroupskip\string~n\string)}}%
```

Define letter group headings.

```
4153 \write\glswrite{^^J; letter group headings^^J}%
4154 \write\glswrite{(markup-letter-group
4155 :open-head \string\string\glsgroupheading
4156 \glsopenbrace\string"^^J\space\space\space
4157 :close-head \string\glsclosebrace\string"}}%
```

Define additional letter groups.

```
4158 \write\glswrite{^^J; additional letter groups^^J}%
4159 \write\glswrite{\@xdylettergroups}}%
```

Define additional sort rules

```
4160 \write\glswrite{^^J; additional sort rules^^J}
4161 \write\glswrite{\@dysortrules}}%
```

Close the style file

```
4162 \closeout\glswrite
```

Suppress any further calls.

```
4163 \let\writeist\relax
4164 }
4165 \else
```

Code to use if makeindex is required.

```

4166 \edef\@gls@actualchar{\string?}
4167 \edef\@gls@encapchar{\string|}
4168 \edef\@gls@levelchar{\string!}
4169 \edef\@gls@quotechar{\string"}
4170 \def\writeist{\relax
4171 \openout\glswrite=\istfilename
4172 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
4173 created by the glossaries package}
4174 \write\glswrite{\expandafter\@gobble\string\% for document
4175 '\jobname' on \the\year-\the\month-\the\day}
4176 \write\glswrite{actual '\@gls@actualchar'}
4177 \write\glswrite{encap '\@gls@encapchar'}
4178 \write\glswrite{level '\@gls@levelchar'}
4179 \write\glswrite{quote '\@gls@quotechar'}
4180 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4181 \write\glswrite{preamble \string"\string\glossarysection[\string
4182 \glossarytoctitle]{\string\glossarytitle}\string
4183 \glossarypreamble\string\n\string\begin{theglossary}\string
4184 \glossaryheader\string\n\string"}
4185 \write\glswrite{postamble \string"\string%\string\n\string
4186 \end{theglossary}\string\glossarypostamble\string\n
4187 \string"}
4188 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
4189 \string"}
4190 \write\glswrite{item_0 \string"\string%\string\n\string"}
4191 \write\glswrite{item_1 \string"\string%\string\n\string"}
4192 \write\glswrite{item_2 \string"\string%\string\n\string"}
4193 \write\glswrite{item_01 \string"\string%\string\n\string"}
4194 \write\glswrite{item_x1
4195 \string"\string\relax \string\glresetentrylist\string\n
4196 \string"}
4197 \write\glswrite{item_12 \string"\string%\string\n\string"}
4198 \write\glswrite{item_x2
4199 \string"\string\relax \string\glresetentrylist\string\n
4200 \string"}

4201 \write\glswrite{delim_0 \string"\string\{\string
4202 \glossaryentrynumbers\string\{\string\relax \string"}
4203 \write\glswrite{delim_1 \string"\string\{\string
4204 \glossaryentrynumbers\string\{\string\relax \string"}
4205 \write\glswrite{delim_2 \string"\string\{\string
4206 \glossaryentrynumbers\string\{\string\relax \string"}
4207 \write\glswrite{delim_t \string"\string\}\string\}\string"}
4208 \write\glswrite{delim_n \string"\string\delimN \string"}
4209 \write\glswrite{delim_r \string"\string\delimR \string"}
4210 \write\glswrite{headings_flag 1}
4211 \write\glswrite{heading_prefix
4212 \string"\string\glsgroupheading\string\{\string"}
4213 \write\glswrite{heading_suffix

```

```

4214      \string\string\}\string\relax
4215      \string\glsresetentrylist \string"}
4216      \write\glswrite{symhead_positive \string"glssymbols\string"}
4217      \write\glswrite{numhead_positive \string"glsnnumbers\string"}
4218      \write\glswrite{page_compositor \string"glscpositor\string"}
4219      \@gls@escbsdq\gls@suffixF
4220      \@gls@escbsdq\gls@suffixFF
4221      \ifx\gls@suffixF\@empty
4222      \else
4223        \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
4224      \fi
4225      \ifx\gls@suffixFF\@empty
4226      \else
4227        \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
4228      \fi
4229      \closeout\glswrite
4230      \let\writeist\relax
4231    }
4232 \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`

```

4233 \newcommand{\noist}{%
    Update attributes list
4234   \@gls@addpredefinedattributes
4235   \let\writeist\relax
4236 }

```

`\@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`

```

4237 \newcommand*{\@makeglossary}[1]{%
4238   \ifglossaryexists{#1}%
4239   {%

```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```

4240 \ifglssavewrites
4241 \expandafter\newtoks\csname glo@#1@filetok\endcsname
4242 \else
4243 \expandafter\newwrite\csname glo@#1@file\endcsname
4244 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
4245 \fi
4246 \@gls@renewglossary
4247 \writeist
4248 }%
4249 {%
4250 \PackageError{glossaries}%
4251 {Glossary type ‘#1’ not defined}%
4252 {New glossaries must be defined before using \string\makeglossary}%
4253 }%
4254 }

```

`\@glsopenfile` Open write file associated with the given glossary.

```

4255 \newcommand*{\@glsopenfile}[2]{%
4256 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
4257 \PackageInfo{glossaries}{Writing glossary file
4258 \jobname.\csname @glotype@#2@out\endcsname}%
4259 }

```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4260 \newcommand*{\@nomakeglossaries}{%
4261 \GlossariesWarningNoLine{\string\makeglossaries\space
4262 hasn't been used,^^Jthe glossaries will not be updated}%
4263 }

```

`\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`

```

4264 \newcommand*{\makeglossaries}{%

```

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.

```

4265 \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{} }
4266 \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{} }
4267 % Write the name of the style file to the aux file
4268 % (needed by \app{makeglossaries})
4269 % \begin{macrocode}
4270 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
4271 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}

```

Iterate through each glossary type and activate it.

```

4272 \@for\@glo@type:=\@glo@types\do{%
4273   \ifthenelse{\equal{\@glo@type}{}}{ }{%
4274     \@makeglossary{\@glo@type}}%
4275 }%

```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```

4276 \renewcommand*\newglossary[4] []{%
4277   \PackageError{glossaries}{New glossaries
4278     must be created before \string\makeglossaries}{You need
4279     to move \string\makeglossaries\space after all your
4280     \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

4281 \let\@makeglossary\relax
4282 \let\makeglossary\relax
4283 \let\makeglossaries\relax

```

Disable all commands that have no effect after `\makeglossaries`

```

4284 \@disable@onlypremakeg

```

Allow see key:

```

4285 \let\gls@checkseeallowed\relax

```

Suppress warning about no `\makeglossaries`

```

4286 \let\warn@nomakeglossaries\relax

```

Declare list parser for `\glsdisplaynumberlist`

```

4287 \ifglssavenumberlist
4288   \edef\@gls@dodeflistparser{\noexpand\DeclareListParser
4289     {\noexpand\glsnumlistparser}{\delimN}}%
4290   \@gls@dodeflistparser
4291 \fi
4292 }

```

Must occur in the preamble:

```

4293 \@onlypreamble{\makeglossaries}

```

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`

```

4294 \let\makeglossary\makeglossaries

```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```

4295 \AtEndDocument{%
4296   \warn@nomakeglossaries
4297   \warn@noprintglossary
4298 }

```

1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```
4299 \newwrite\glswrite
```

`\istfile` Deprecated.

```
4300 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```
4301 \AtEndDocument{%
```

```
4302   \glswritefiles
```

```
4303 }
```

`\@glswritefiles` Only write the files if `savewrites=true`

```
4304 \newcommand*{\@glswritefiles}{%
```

Iterate through all the glossaries

```
4305   \forallglossaries{\@glo@type}{%
```

Check for empty glossaries (patch provided by Patrick Häcker)

```
4306     \ifcsundef{glo@\@glo@type @filetok}%
```

```
4307     {%
```

```
4308       \def\gls@tmp{}%
```

```
4309     }%
```

```
4310     {%
```

```
4311       \edef\gls@tmp{\expandafter\the
```

```
4312         \csname glo@\@glo@type @filetok\endcsname}%
```

```
4313     }%
```

```
4314     \ifx\gls@tmp\@empty
```

```
4315       \ifx\@glo@type\glsdefaulttype
```

```
4316         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
4317           entries.^^JRemember to use package option ‘nomain’ if
```

```
4318 you
```

```
4319           don’t want to^^Juse the main glossary}%
```

```
4320       \else
```

```
4321         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
4322           entries}%
```

```
4323       \fi
```

```
4324     \else
```

```
4325       \@glsopenfile{\glswrite}{\@glo@type}%
```

```
4326       \immediate\write\glswrite{%
```

```
4327         \expandafter\the
```

```
4328         \csname glo@\@glo@type @filetok\endcsname}%
```

```
4329       \immediate\closeout\glswrite
```

```
4330     \fi
```

```
4331   }%
```

```
4332 }
```

The `\glossary` command is redefined so that it takes an optional argument `<type>` to specify the glossary type (use `\glsdefaulttype` glossary by default).

This shouldn't be used at user level as `\glslink` sets the correct format. The associated number should be stored in `\theglsentrycounter` before using `\glossary`.

`\glossary`

```
4333 \renewcommand*{\glossary}[1][\glsdefaultttype]{%
4334   \@glossary[#1]%
4335 }
```

Define internal `\@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.)

`\@glossary`

```
4336 \def\@glossary[#1]{\index}
```

This is a convenience command to set `\@glossary`. It is used by `\@makeglossary` and then redefined to do nothing, as it only needs to be done once.

`\@gls@renewglossary`

```
4337 \newcommand{\@gls@renewglossary}{%
4338   \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
4339   \let\@gls@renewglossary\empty
4340 }
```

The `\@wrglossary` command is redefined 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`).

`\@wrglossary`

```
4341 \renewcommand*{\@wrglossary}[2]{%
4342   \ifglssavewrites
4343     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4344     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4345       \expandafter{\@gls@tmp^^J}%
4346   \else
4347     \ifcsdef{glo@#1@file}%
4348       {%
4349         \expandafter\protected@write\csname glo@#1@file\endcsname{%
4350           \gls@disablepagerefexpansion}{#2}%
4351       }%
4352     {%
4353       \GlossariesWarning{No file defined for glossary ‘#1’}%
4354     }%
4355   \fi
4356 \endgroup\@esphack
4357 }
```

```

\do@wrglossary
4358 \newcommand*\do@wrglossary}[1]{%
4359   \ifglindexonlyfirst
4360     \ifglused{#1}{\do@wrglossary{#1}}%
4361   \else
4362     \do@wrglossary{#1}%
4363   \fi
4364 }

```

`@protected@pagefmts` List of page formats to be protected against expansion.

```

4365 \newcommand*\gls@protected@pagefmts{%
4366   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4367 }

```

`blepagerefexpansion`

```

4368 \newcommand*\gls@disablepagerefexpansion{%
4369   \@for\@gls@this:=\gls@protected@pagefmts\do
4370   {%
4371     \expandafter\let\@gls@this\relax
4372   }%
4373 }

```

`\gls@alphpage`

```

4374 \newcommand*\gls@alphpage{\@alph\c@page}

```

`\gls@Alphpage`

```

4375 \newcommand*\gls@Alphpage{\@Alph\c@page}

```

`\gls@numberpage`

```

4376 \newcommand*\gls@numberpage{\number\c@page}

```

`\gls@romanpage`

```

4377 \newcommand*\gls@romanpage{\romannumeral\c@page}

```

`\gls@Romanpage`

```

4378 \newcommand*\gls@Romanpage{\@Roman\c@page}

```

`\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.

```

4379 \newcommand*\do@wrglossary}[1]{%
4380   \begingroup

```

First a bit of hackery to prevent premature expansion of `\c@page`. Store original definitions:

```

4381   \let\orgthe\the
4382   \let\orgnumber\number
4383   \let\orgromannumeral\romannumeral
4384   \let\orgalph\@alph
4385   \let\orgAlph\@Alph
4386   \let\orgRoman\@Roman

```


Redefine:

```
4387 \def\the##1{%
4388 \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4389 \def\number##1{%
4390 \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4391 \def\romannumeral##1{%
4392 \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4393 \def\@Roman##1{%
4394 \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4395 \def\@alph##1{%
4396 \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4397 \def\@Alph##1{%
4398 \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%
```

Prevent expansion:

```
4399 \gls@disablepagerefexpansion
```

Now store location in \@glslocref:

```
4400 \protected@xdef\@glslocref{\theglentrycounter}%
4401 \endgroup
```

Escape any special characters

```
4402 \@gls@checkmkidxchars\@glslocref
```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
4403 \expandafter\ifx\theHglentrycounter\theglentrycounter
4404 \def\@glo@counterprefix{%
4405 \else
4406 \protected@edef\@glsHlocref{\theHglentrycounter}%
4407 \@gls@checkmkidxchars\@glsHlocref
4408 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4409 {\@glslocref}{\@glsHlocref}%
4410 }%
4411 \@do@gls@getcounterprefix
4412 \fi
```

De-tok label if required

```
4413 \edef\@gls@label{\glsdetoklabel{#1}}%
```

Determine whether to use xindy or makeindex syntax

```
4414 \ifglsxindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```
4415 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
4416 \def\@glo@range{%
4417 \expandafter\if\@glo@prefix(\relax
4418 \def\@glo@range{:open-range}%
4419 \else
4420 \expandafter\if\@glo@prefix)\relax
4421 \def\@glo@range{:close-range}%
```

```

4422     \fi
4423     \fi

Write to the glossary file using xindy syntax.

4424     \glossary[\csname glo@\gls@label @type\endcsname]{%
4425     (indexentry :tkey (\csname glo@\gls@label @index\endcsname)

4426         :locref \string"{\@glo@counterprefix}{\@gls@locref}\string" %
4427         :attr \string"\@gls@counter\@glo@suffix\string"
4428         \@glo@range
4429     )
4430     }%
4431     \else

Convert the format information into the format required for makeindex

4432     \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@gls@numberformat}%
4433     {\@glo@counterprefix}%

Write to the glossary file using makeindex syntax.

4434     \glossary[\csname glo@\gls@label @type\endcsname]{%
4435     \string\glossaryentry{\csname glo@\gls@label @index\endcsname
4436         \@gls@encapchar\@glo@numfmt}{\@gls@locref}}%
4437     \fi
4438 }

```

`\gls@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.)

```

4439 \newcommand*\@gls@getcounterprefix[2]{%
4440     \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4441     \ifx\@gls@thisloc\@gls@thisHloc
4442         \def\@glo@counterprefix{%
4443         \else
4444             \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4445                 \def\@glo@tmp{##2}%
4446                 \ifx\@glo@tmp\@empty
4447                     \def\@glo@counterprefix{%
4448                     \else
4449                         \def\@glo@counterprefix{##1}%
4450                 \fi
4451             }%
4452             \@gls@get@counterprefix#2.#1\end@getprefix
4453         \fi
4454 }

```

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.

```

4455 \newcommand{\do@seeglossary}[2]{%
4456 \def\@gls@xref{#2}%
4457 \@onelevel@sanitize\@gls@xref
4458 \@gls@checkmkidxchars\@gls@xref
4459 \ifglxsindy
4460   \glossary[\csname glo@#1@type\endcsname]{%
4461     (indexentry
4462       :key (\csname glo@#1@index\endcsname)
4463       :xref (\string"\@gls@xref\string")
4464       :attr \string"see\string"
4465     )
4466   }%
4467 \else
4468   \glossary[\csname glo@#1@type\endcsname]{%
4469     \string\glossaryentry{\csname glo@#1@index\endcsname
4470       \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4471 \fi
4472 }
```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4473 \def\@gls@fixbraces#1#2#3\@nil{%
4474   \ifx#2[\relax
4475     \def#1{#2#3}%
4476   \else
4477     \def#1{\{#2#3\}}%
4478   \fi
4479 }
```

`\glssee` `\glssee{\langle label \rangle} \{ \langle cross-ref list \rangle \}`

```

4480 \DeclareRobustCommand*\glssee}[3][\seename]{%
4481   \do@seeglossary{#2}{[#1]{#3}}
4482 \newcommand*\@glssee}[3][\seename]{%
4483   \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.

```

4484 \DeclareRobustCommand*\glsseeformat}[3][\seename]{%
4485   \emph{#1} \glsseelist{#2}}
```

`\glsseelist` `\glsseelist{\langle list \rangle}` formats list of entry labels.

```

4486 \DeclareRobustCommand*\glsseelist}[1]{%
```

If there is only one item in the list, set the last separator to do nothing.

```

4487   \let\@gls@dolast\relax
```

Don't display separator on the first iteration of the loop

```
4488 \let\@gls@donext\relax
```

Iterate through the labels

```
4489 \@for\@gls@thislabel:=#1\do{%
```

Check if on last iteration of loop

```
4490 \ifx\@xfor@nextelement\@nnil
4491 \@gls@dolast
4492 \else
4493 \@gls@donext
4494 \fi
```

Display the entry for this label. (Expanding label as it's a temporary control sequence that's used elsewhere.)

```
4495 \expandafter\glsseeitem\expandafter{\@gls@thislabel}%
```

Update separators

```
4496 \let\@gls@dolast\glsseelastsep
4497 \let\@gls@donext\glsseesep
4498 }%
4499 }
```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4500 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entries in a cross-referencing list.

```
4501 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4502 \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.)

```
4503 \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.

```
4504 \newcommand*{\gls@save@numberlist}[1]{%
4505 \ifglssavenumberlist
4506 \toks@{#1}%
```

```

4507 \edef\@do@writeaux@info{%
4508 \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
4509 }%
4510 \@onelevel@sanitize\@do@writeaux@info
4511 \protected@write\@auxout{}\{\@do@writeaux@info}%
4512 \fi
4513 }

```

`\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.)

```

4514 \def\warn@noprintglossary{%
4515 \GlossariesWarningNoLine{No \string\printglossary\space
4516 or \string\printglossaries\space
4517 found.^^JThis document will not have a glossary}%
4518 }

```

`\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.

```

4519 \ifcsundef{printglossary}{}%
4520 {%

```

If `\printglossary` is already defined, issue a warning and undefine it.

```

4521 \GlossariesWarning{Overriding \string\printglossary}%
4522 \undef\printglossary
4523 }

```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```

4524 \newcommand*{\printglossary}[1][type=\glstdefaulttype]{%

```

Set up defaults.

```

4525 \def\@glo@type{\glstdefaulttype}%
4526 \def\glossarytitle{\csname @glo@type@\@glo@type @title\endcsname}%

4527 \def\glossarytoctitle{\glossarytitle}%
4528 \let\org@glossarytitle\glossarytitle
4529 \def\@glossarystyle{}%
4530 \def\gls@dotoc@title{\glstsettoctitle{\@glo@type}}%

```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```

4531 \let\@org@glossaryentrynumbers\glossaryentrynumbers

```

Localise the effects of the optional argument

```

4532 \bgroup

```

Determine settings specified in the optional argument.

```

4533 \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)

```
4534 \ifx\glossarytitle\org@glossarytitle
4535 \else
4536 \expandafter\let\csname @glo@type\@glo@type @title\endcsname
4537 \glossarytitle
4538 \fi
```

Allow a high-level user command to indicate the current glossary

```
4539 \let\currentglossary\@glo@type
```

Enable individual number lists to be suppressed.

```
4540 \let\org@glossaryentrynumbers\glossaryentrynumbers
4541 \let\glsnonextpages\glsnonextpages
```

Enable individual number list to be activated:

```
4542 \let\glsnextpages\glsnextpages
```

Enable suppression of description terminators.

```
4543 \let\nopostdesc\nopostdesc
```

Set up the entry for the TOC

```
4544 \gls@dotoc@title
```

Set the glossary style

```
4545 \@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):

```
4546 \let\gls@org@glossaryentryfield\glossentry
4547 \let\gls@org@glossarysubentryfield\subglossentry
4548 \renewcommand{\glossentry}[1]{%
4549 \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4550 \gls@org@glossaryentryfield{##1}%
4551 }%
4552 \renewcommand{\subglossentry}[2]{%
4553 \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4554 \gls@org@glossarysubentryfield{##1}{##2}%
4555 }%
```

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.)

```
4556 \makeatletter
```

Input the glossary file, if it exists.

```
4557 \@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.

```

4558 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4559 {}%
4560 {\null}%

```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

4561 \ifglxindy
4562 \ifcsundef{@xdy@\@glo@type @language}%
4563 {%
4564 \edef\@do@auxoutstuff{%
4565 \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.

```

4566 \noexpand\immediate\noexpand\write\@auxout{%
4567 \string\providecommand\string\@xdylanguage[2]{}}%
4568 \noexpand\immediate\noexpand\write\@auxout{%
4569 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4570 }%
4571 }%
4572 }%
4573 {%
4574 \edef\@do@auxoutstuff{%
4575 \noexpand\AtEndDocument{%
4576 \noexpand\immediate\noexpand\write\@auxout{%
4577 \string\providecommand\string\@xdylanguage[2]{}}%
4578 \noexpand\immediate\noexpand\write\@auxout{%
4579 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4580 @language\endcsname}}%
4581 }%
4582 }%
4583 }%
4584 \@do@auxoutstuff
4585 \edef\@do@auxoutstuff{%
4586 \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.

```

4587 \noexpand\immediate\noexpand\write\@auxout{%
4588 \string\providecommand\string\@gls@codepage[2]{}}%
4589 \noexpand\immediate\noexpand\write\@auxout{%
4590 \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
4591 }%
4592 }%
4593 \@do@auxoutstuff
4594 \fi
4595 \egroup

```

```

Reset \glossaryentrynumbers
4596 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
Suppress warning about no \printglossary
4597 \global\let\warn@noprintglossary\relax
4598 }

```

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`

```

4599 \newcommand*{\printglossaries}{%
4600 \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
4601 }

```

The keys that can be used in the optional argument to `\printglossary` are as follows: The `type` key sets the glossary type.

```

4602 \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`.

```

4603 \define@key{printgloss}{title}{%
4604 \def\glossarytitle{#1}%
4605 \let\gls@dotoc@title\relax
4606 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

4607 \define@key{printgloss}{toctitle}{%
4608 \def\glossarytoctitle{#1}%
4609 \let\gls@dotoc@title\relax
4610 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

4611 \define@key{printgloss}{style}{%
4612 \ifcsundef{@glsstyle@#1}%
4613 {%
4614 \PackageError{glossaries}%
4615 {Glossary style ‘#1’ undefined}{}%
4616 }%
4617 {%
4618 \def\@glossarystyle{\setglossentrycompatibility
4619 \csname @glsstyle@#1\endcsname}%
4620 }%
4621 }

```


The `numberedsection` key determines if this glossary should be in a numbered section.

```

4622 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
4623 false,nolabel,autolabel,nameref}[nolabel]{%
4624   \ifcase\nr\relax
4625     \renewcommand*{\@@glossarysecstar}{*}%
4626     \renewcommand*{\@@glossaryseclabel}{}%
4627   \or
4628     \renewcommand*{\@@glossarysecstar}{}%
4629     \renewcommand*{\@@glossaryseclabel}{}%
4630   \or
4631     \renewcommand*{\@@glossarysecstar}{}%
4632     \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
4633   \or
4634     \renewcommand*{\@@glossarysecstar}{*}%
4635     \renewcommand*{\@@glossaryseclabel}{%
4636       \protected@edef\@currentlabelname{\glossarytoctitle}%
4637       \label{\glsautoprefix\@glo@type}}%
4638   \fi
4639 }
```

The `nogroupskip` key determines whether or not there should be a vertical gap between glossary groups.

```

4640 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
4641   \csuse{glsnogroupskip#1}%
4642 }
```

The `nonumberlist` key determines if this glossary should have a number list.

```

4643 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
4644   \ifglsnonumberlist
4645     \def\glossaryentrynumbers##1{%
4646   \else
4647     \def\glossaryentrynumbers##1{##1}%
4648   \fi}
```

`\@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.

```

4649 \newcommand*{\@glsnonextpages}{%
4650   \gdef\glossaryentrynumbers##1{%
4651     \glsresetentrylist
4652   }%
4653 }
```

`\@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.

```
4654 \newcommand*{\@glsnextpages}{%
4655   \gdef\glossaryentrynumbers##1{%
4656     ##1\glsresetentrylist}}
```

\glsresetentrylist Resets \glossaryentrynumbers

```
4657 \newcommand*{\glsresetentrylist}{%
4658   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}
```

\glsnonextpages Outside of \printglossary this does nothing.

```
4659 \newcommand*{\glsnonextpages}{}%
```

\glsnextpages Outside of \printglossary this does nothing.

```
4660 \newcommand*{\glsnextpages}{}%
```

glossaryentry If the entrycounter package option has been used, define a counter to number each level 0 entry.

```
4661 \ifglentrycounter
4662   \ifx\@gls@counterwithin\@empty
4663     \newcounter{glossaryentry}
4664   \else
4665     \newcounter{glossaryentry}[\@gls@counterwithin]
4666   \fi
4667   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
4668 \fi
```

glossarysubentry If the subentrycounter package option has been used, define a counter to number each level 1 entry.

```
4669 \ifglssubentrycounter
4670   \ifglentrycounter
4671     \newcounter{glossarysubentry}[glossaryentry]
4672   \else
4673     \newcounter{glossarysubentry}
4674   \fi
4675   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
4676 \fi
```

esetsubentrycounter Resets the glossarysubentry counter.

```
4677 \ifglssubentrycounter
4678   \newcommand*{\glsresetsubentrycounter}{%
4679     \setcounter{glossarysubentry}{0}%
4680   }
4681 \else
4682   \newcommand*{\glsresetsubentrycounter}{}%
4683 \fi
```

`\resetsubentrycounter` Resets the glossaryentry counter.

```
4684 \ifglentrycounter
4685   \newcommand*{\glsresetentrycounter}{%
4686     \setcounter{glossaryentry}{0}%
4687   }
4688 \else
4689   \newcommand*{\glsresetentrycounter}{}
4690 \fi
```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```
4691 \ifglentrycounter
4692   \newcommand*{\glsstepentry}[1]{%
4693     \refstepcounter{glossaryentry}%
4694     \label{glentry-\glsdetoklabel{#1}}%
4695   }
4696 \else
4697   \newcommand*{\glsstepentry}[1]{}
4698 \fi
```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```
4699 \ifglssubentrycounter
4700   \newcommand*{\glsstepsubentry}[1]{%
4701     \edef\currentglssubentry{\glsdetoklabel{#1}}%
4702     \refstepcounter{glossarysubentry}%
4703     \label{glentry-\currentglssubentry}%
4704   }
4705 \else
4706   \newcommand*{\glsstepsubentry}[1]{}
4707 \fi
```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```
4708 \ifglentrycounter
4709   \newcommand*{\glsrefentry}[1]{\ref{glentry-\glsdetoklabel{#1}}}
4710 \else
4711   \ifglssubentrycounter
4712     \newcommand*{\glsrefentry}[1]{\ref{glentry-\glsdetoklabel{#1}}}
4713   \else
4714     \newcommand*{\glsrefentry}[1]{\gls{#1}}
4715   \fi
4716 \fi
```

`\glsentrycounterlabel` Defines how to display the glossaryentry counter.

```
4717 \ifglentrycounter
4718   \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
4719 \else
4720   \newcommand*{\glsentrycounterlabel}{}
4721 \fi
```

`subentrycounterlabel` Defines how to display the `glossarysubentry` counter.

```
4722 \ifglssubentrycounter
4723   \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry}\space}
4724 \else
4725   \newcommand*{\glssubentrycounterlabel}{}
4726 \fi
```

`\glsenentryitem` Step and display `glossaryentry` counter, if appropriate.

```
4727 \ifglsenentrycounter
4728   \newcommand*{\glsenentryitem}[1]{%
4729     \glssstepentry{#1}\glsenentrycounterlabel
4730   }
4731 \else
4732   \newcommand*{\glsenentryitem}[1]{\glssresetsubentrycounter}
4733 \fi
```

`\glsssubentryitem` Step and display `glossarysubentry` counter, if appropriate.

```
4734 \ifglsssubentrycounter
4735   \newcommand*{\glsssubentryitem}[1]{%
4736     \glssstepsubentry{#1}\glsssubentrycounterlabel
4737   }
4738 \else
4739   \newcommand*{\glsssubentryitem}[1]{}
4740 \fi
```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by `glossary` styles.

```
4741 \ifcsundef{theglossary}%
4742 {%
4743   \newenvironment{theglossary}{}{}}%
4744 }%
4745 {%
4746   \GlossariesWarning{overriding ‘theglossary’ environment}%
4747   \renewenvironment{theglossary}{}{}}%
4748 }
```

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`

```
4749 \newcommand*{\glossaryheader}{}%
```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```
4750 \newcommand*{\glstarget}[2]{\@glstarget{\glolinkprefix#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}`

```
4751 \providecommand*{\compatibleglossentry}[2]{%
4752   \toks@{#2}%
4753   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
4754     {\noexpand\glsnamefont
4755       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
4756     {\expandafter\expandonce\csname glo@#1@desc\endcsname}}%
4757     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}}%
4758     {\the\toks@}%
4759   }%
4760   \@do@glossentry
4761 }
```

`\glossentryname`

```
4762 \newcommand*{\glossentryname}[1]{%
4763   \glsdoifexistsorwarn{#1}%
4764   {%
4765     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
4766     \expandafter\glsnamefont\expandafter{\glo@name}%
4767   }%
4768 }
```

`\Glossentryname`

```
4769 \newcommand*{\Glossentryname}[1]{%
4770   \glsdoifexistsorwarn{#1}%
4771   {%
4772     \glsnamefont{\Glsentryname{#1}}%
4773   }%
4774 }
```

`\glossentrydesc`

```
4775 \newcommand*{\glossentrydesc}[1]{%
4776   \glsdoifexistsorwarn{#1}%
4777   {%
4778     \glsentrydesc{#1}%
4779   }%
4780 }
```

\Glossentrydesc

```
4781 \newcommand*{\Glossentrydesc}[1]{%
4782   \glsdoifexistsorwarn{#1}%
4783   {%
4784     \Glsentrydesc{#1}%
4785   }%
4786 }
```

\glossentrysymbol

```
4787 \newcommand*{\glossentrysymbol}[1]{%
4788   \glsdoifexistsorwarn{#1}%
4789   {%
4790     \glsentrysymbol{#1}%
4791   }%
4792 }
```

\Glossentrysymbol

```
4793 \newcommand*{\Glossentrysymbol}[1]{%
4794   \glsdoifexistsorwarn{#1}%
4795   {%
4796     \Glsentrysymbol{#1}%
4797   }%
4798 }
```

patiblesubglossentry

\subglossentry{<level>}{<label>}{<page-list>}

```
4799 \providecommand*{\compatiblesubglossentry}[3]{%
4800   \toks@{#3}%
4801   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
4802     {#2}%
4803     {\noexpand\glsnamefont
4804       {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
4805     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
4806     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
4807     {\the\toks@}}%
4808   }%
4809   \@do@subglossentry
4810 }
```

sentrycompatibility

```
4811 \newcommand*{\setglossentrycompatibility}{%
4812   \let\glossentry\compatibleglossentry
4813   \let\subglossentry\compatiblesubglossentry
4814 }
4815 \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.

```
4816 \newcommand{\glossaryentryfield}[5]{%
4817   \GlossariesWarning
4818   {Deprecated use of \string\glossaryentryfield.^^J
4819     I recommend you change to \string\glossentry.^^J
4820     If you've just upgraded, try removing your gls auxiliary
4821     files^^J and recompile}%
4822   \noindent\textbf{\glstarget{#1}{#2}} #4 #3. #5\par}
```

`\glossarysubentryfield`

`\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.)

```
4823 \newcommand*{\glossarysubentryfield}[6]{%
4824   \GlossariesWarning
4825   {Deprecated use of \string\glossarysubentryfield.^^J
4826     I recommend you change to \string\subglossentry.^^J
4827     If you've just upgraded, try removing your gls auxiliary
4828     files^^J and recompile}%
4829   \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`

```
4830 \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: `glssymbols`, `glsnumbers`, A, ..., Z. Glossary styles must

redefined this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
4831 \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 `\glssymbolsgroupname`, 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`

```
4832 \newcommand*{\glsgetgrouptitle}[1]{%
4833   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
4834   \@gls@grptitle
4835 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
4836 \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.

```
4837 \dtl@ifsingle{#1}%
4838 {%
4839   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
4840 }%
4841 {%
4842   \ifboolexpr{test{\ifstrequal{#1}{glsymbols}}
4843               or test{\ifstrequal{#1}{glsnumbers}}}%
4844   {%
4845     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
```



```

4846 }%
4847 {%
4848     \def#2{#1}%
4849 }%
4850 }%
4851 }

```

`\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 `\glssetgrouptitle`, you will also need to redefine `\glsgetgrouplabel`.

`\glsgetgrouplabel`

```

4852 \newcommand*{\glsgetgrouplabel}[1]{%
4853 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
4854 \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`

```

4855 \newcommand*{\setentrycounter}[2][1]{%
4856   \def\@glo@counterprefix{#1}%
4857   \ifx\@glo@counterprefix\@empty
4858     \def\@glo@counterprefix{.}%
4859   \else
4860     \def\@glo@counterprefix{.#1.}%
4861   \fi
4862   \def\glsentrycounter{#2}%
4863 }

```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```

4864 \newcommand*{\setglossarystyle}[1]{%
4865   \ifcsundef{@glsstyle@#1}%
4866   {%
4867     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4868   }%
4869   {%
4870     \csname @glsstyle@#1\endcsname
4871   }%
4872 }

```

`\glossarystyle`

```

4873 \newcommand*{\glossarystyle}[1]{%

```

```

4874 \ifcsundef{@glsstyle@#1}%
4875 {%
4876   \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4877 }%
4878 {%
4879   \GlossariesWarning
4880   {Deprecated command \string\glossarystyle.^^J
4881     I recommend you switch to \string\setglossarystyle\space unless
4882     you want to maintain backward compatibility}%
4883   \setglossentrycompatibility
4884   \csname @glsstyle@#1\endcsname
4885   \ifcsdef{@glscompstyle@#1}%
4886     {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
4887   {}%
4888 }%
4889 }

```

`\newglossarystyle` New glossary styles can be defined using:

```
\newglossarystyle{<name>}{<definition>}
```

The *<definition>* argument should redefine the `\glossary`, `\glossaryheader`, `\glsgroupheading`, `\glossaryentryfield` and `\glsgroupskip` (see [subsection 1.18](#) for the definitions of predefined styles). Glossary styles should not redefine `\glossarypreamble` and `\glossarypostamble`, as the user should be able to switch between styles without affecting the pre- and postambles.

```

4890 \newcommand{\newglossarystyle}[2]{%
4891   \ifcsundef{@glsstyle@#1}%
4892   {%
4893     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
4894   }%
4895   {%
4896     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
4897   }%
4898 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

4899 \newcommand{\renewglossarystyle}[2]{%
4900   \ifcsundef{@glsstyle@#1}%
4901   {%
4902     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
4903   }%
4904   {%
4905     \csdef{@glsstyle@#1}{#2}%
4906   }%
4907 }

```

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`

```
4908 \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 `\glshypernumber`. 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.

`\glshypernumber`

```
4909 \ifcsundef{hyperlink}%
4910 {%
4911   \def\glshypernumber#1{#1}%
4912 }%
4913 {%
4914   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
4915 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
4916 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
4917   \ifx\#1\%
4918     \else
4919       \@delimR#1\delimR\delimR\%
4920     \fi
4921     \ifx\#2\%
4922       \else
4923         #2%
4924       \fi
4925     \ifx\#3\%
4926       \else
4927         \@glshypernumber#3\@nil
4928       \fi
4929 }
```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\gls@hypernumber`).

`\@delimR`

```
4930 \def\@delimR#1\delimR #2\delimR #3\\{%
4931 \ifx\\#2\\%
4932   \@delimN{#1}%
4933 \else
4934   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
4935 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```
4936 \def\@delimN#1{\@delimN#1\delimN \delimN\\}
4937 \def\@delimN#1\delimN #2\delimN#3\\{%
4938 \ifx\\#3\\%
4939   \@gls@numberlink{#1}%
4940 \else
4941   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
4942 \fi
4943 }
```

The following code is modified from `hyperref's \HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```
4944 \def\@gls@numberlink#1{%
4945 \begingroup
4946 \toks@={}%
4947 \@gls@removespaces#1 \@nil
4948 \endgroup}

4949 \def\@gls@removespaces#1 #2\@nil{%
4950 \toks@=\expandafter{\the\toks@#1}%
4951 \ifx\\#2\\%
4952   \edef\x{\the\toks@}%
4953   \ifx\x\empty
4954     \else

4955     \hyperlink{\glsentrycounter\@gls@counterprefix\the\toks@}%
4956     {\the\toks@}%
4957   \fi
4958 \else
4959   \@gls@ReturnAfterFi{%
4960     \@gls@removespaces#2\@nil
4961   }%
4962 \fi
4963 }
4964 \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`
 4965 `\newcommand*{\hyperrm}[1]{\textrm{\glshypernumber{#1}}}`

`\hypersf`
 4966 `\newcommand*{\hypersf}[1]{\textsf{\glshypernumber{#1}}}`

`\hypertt`
 4967 `\newcommand*{\hypertt}[1]{\texttt{\glshypernumber{#1}}}`

`\hyperbf`
 4968 `\newcommand*{\hyperbf}[1]{\textbf{\glshypernumber{#1}}}`

`\hypermd`
 4969 `\newcommand*{\hypermd}[1]{\textmd{\glshypernumber{#1}}}`

`\hyperit`
 4970 `\newcommand*{\hyperit}[1]{\textit{\glshypernumber{#1}}}`

`\hypersl`
 4971 `\newcommand*{\hypersl}[1]{\textsl{\glshypernumber{#1}}}`

`\hyperup`
 4972 `\newcommand*{\hyperup}[1]{\textup{\glshypernumber{#1}}}`

`\hypersc`
 4973 `\newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}`

`\hyperemph`
 4974 `\newcommand*{\hyperemph}[1]{\emph{\glshypernumber{#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.

```

4975 \newcommand{\oldacronym}[4][\gls@label]{%
4976   \def\gls@label{#2}%
4977   \newacronym[#4]{#1}{#2}{#3}%
4978   \ifcsundef{xspace}%
4979     {%
4980       \expandafter\edef\csname#1\endcsname{%
4981         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}%
4982       }%
4983     }%
4984     {%
4985       \expandafter\edef\csname#1\endcsname{%
4986         \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
4987           \noexpand\gls{#1}\noexpand\xspace}%
4988       }%
4989     }%
4990 }
```

`\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`

```

4991 \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 needed to cancel it out.

```
4992 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

```
\glstextup
```

```
4993 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}
```

The following are defined for compatibility with version 2.07 and earlier.

```
\glsshortkey
```

```
4994 \newcommand*{\glsshortkey}{short}
```

```
\glsshortpluralkey
```

```
4995 \newcommand*{\glsshortpluralkey}{shortplural}
```

```
\glslongkey
```

```
4996 \newcommand*{\glslongkey}{long}
```

```
\glslongpluralkey
```

```
4997 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
4998 \newrobustcmd*{\acrfull}{%
```

```
4999 \ifstar\s@acrfull\ns@acrfull
```

```
5000 }
```

```
5001 \newcommand*\s@acrfull[2][{}]{%
```

```
5002 \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}%
```

```
5003 \@acrfull{hyper=false,#1}{#2}[]}%
```

```
5004 }
```

```
5005 \newcommand*\ns@acrfull[2][{}]{%
```

```
5006 \new@ifnextchar[{\@acrfull{#1}{#2}}%
```

```
5007 \@acrfull{#1}{#2}[]}%
```

```
5008 }
```

`\@acrfull` Low-level macro:

```
5009 \def\@acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5010 \acrfullfmt{#1}{#2}{#3}%
```

```
5011 }
```

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.

```
5012 \newcommand*{\acrfullfmt}[3]{%
5013   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
5014 }
```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```
5015 \newcommand{\acrlinkfullformat}[5]{%
5016   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%
5017 }
```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```
5018 \newcommand{\acrfullformat}[2]{#1\space(#2)}
```

Default format for full acronym

`\Acrfull`

```
5019 \newrobustcmd*{\Acrfull}{%
5020   \@ifstar\s@Acrfull\ns@Acrfull
5021 }

5022 \newcommand*\s@Acrfull[2][]{%
5023   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}%
5024     {\@Acrfull{hyper=false,#1}{#2}[]}%
5025 }
5026 \newcommand*\ns@Acrfull[2][]{%
5027   \new@ifnextchar[{\@Acrfull{#1}{#2}}%
5028     {\@Acrfull{#1}{#2}[]}%
5029 }
```

Low-level macro:

```
5030 \def\@Acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5031   \Acrfullfmt{#1}{#2}{#3}%
5032 }
```

`\Acrfullfmt` First letter upper case full format.

```
5033 \newcommand*{\Acrfullfmt}[3]{%
5034   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
5035 }
```

`\ACRfull`

```
5036 \newrobustcmd*{\ACRfull}{%
5037   \@ifstar\s@ACRfull\ns@ACRfull
5038 }
```



```

5039 \newcommand*\s@ACRfull[2][]{%
5040   \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}{%
5041     {\@ACRfull{hyper=false,#1}{#2}[]}%
5042 }
5043 \newcommand*\ns@ACRfull[2][]{%
5044   \new@ifnextchar[{\@ACRfull{#1}{#2}}{%
5045     {\@ACRfull{#1}{#2}[]}%
5046 }

```

Low-level macro:

```
5047 \def\@ACRfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```

5048   \ACRfullfmt{#1}{#2}{#3}%
5049 }

```

`\ACRfullfmt` All upper case full format.

```

5050 \newcommand*\@ACRfullfmt[3]{%
5051   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
5052 }

```

Plural:

`\acrfullpl`

```

5053 \newrobustcmd*\acrfullpl{%
5054   \@ifstar\s@acrfullpl\ns@acrfullpl
5055 }

5056 \newcommand*\s@acrfullpl[2][]{%
5057   \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}{%
5058     {\@acrfullpl{hyper=false,#1}{#2}[]}%
5059 }
5060 \newcommand*\ns@acrfullpl[2][]{%
5061   \new@ifnextchar[{\@acrfullpl{#1}{#2}}{%
5062     {\@acrfullpl{#1}{#2}[]}%
5063 }

```

Low-level macro:

```
5064 \def\@acrfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```

5065   \acrfullplfmt{#1}{#2}{#3}%
5066 }

```

`\acrfullplfmt` No case change plural full format.

```

5067 \newcommand*\@acrfullplfmt[3]{%
5068   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5069 }

```

\Acrfullpl

```
5070 \newrobustcmd*{\Acrfullpl}{%
5071   \@ifstar\s@Acrfullpl\ns@Acrfullpl
5072 }

5073 \newcommand*\s@Acrfullpl[2][]{%
5074   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
5075     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
5076 }
5077 \newcommand*\ns@Acrfullpl[2][]{%
5078   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
5079     {\@Acrfullpl{#1}{#2}[]}%
5080 }
```

Low-level macro:

```
5081 \def\@Acrfullpl#1#2[#3]{%
    Make it easier for acronym styles to change this:
5082   \Acrfullplfmt{#1}{#2}{#3}%
5083 }
```

\Acrfullplfmt First letter upper case plural full format.

```
5084 \newcommand*{\Acrfullplfmt}[3]{%
5085   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5086 }
```

\ACRfullpl

```
5087 \newrobustcmd*{\ACRfullpl}{%
5088   \@ifstar\s@ACRfullpl\ns@ACRfullpl
5089 }

5090 \newcommand*\s@ACRfullpl[2][]{%
5091   \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
5092     {\@ACRfullpl{hyper=false,#1}{#2}[]}%
5093 }
5094 \newcommand*\ns@ACRfullpl[2][]{%
5095   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
5096     {\@ACRfullpl{#1}{#2}[]}%
5097 }
```

Low-level macro:

```
5098 \def\@ACRfullpl#1#2[#3]{%
    Make it easier for acronym styles to change this:
5099   \ACRfullplfmt{#1}{#2}{#3}%
5100 }
```

\ACRfullplfmt All upper case plural full format.

```
5101 \newcommand*{\ACRfullplfmt}[3]{%
5102   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
5103 }
```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
5104 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
5105 \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.

```
5106 \newcommand*{\acrnameformat}[2]{\acronymfont{#1}}
```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```
5107 \newtoks\glskeylisttok
```

`\glslabeltok`

```
5108 \newtoks\glslabeltok
```

`\glsshorttok`

```
5109 \newtoks\glsshorttok
```

`\glslongtok`

```
5110 \newtoks\glslongtok
```

`\newacronymhook` Provide a hook for `\newacronym`:

```
5111 \newcommand*{\newacronymhook}{}
```

`\setGenericNewAcronym` New improved version of setting the acronym style.

```
5112 \newcommand*{\SetGenericNewAcronym}{%
5113   \renewcommand{\newacronym}[4][\]{%
5114     \ifdefempty{\@glsacronymlists}%
5115     {%
5116       \def\@glo@type{\acronymtype}%
5117       \setkeys{glossentry}{##1}%
5118       \DeclareAcronymList{\@glo@type}%
5119     }%
5120   }%
5121   \glskeylisttok{##1}%
5122   \glslabeltok{##2}%
5123   \glsshorttok{##3}%
5124   \glslongtok{##4}%
5125   \newacronymhook
5126   \protected@edef\@do@newglossaryentry{%
5127     \noexpand\newglossaryentry{\the\glslabeltok}%
5128     {%
5129       type=\acronymtype,%
```

```

5130     name={\expandonce{\acronymentry{##2}}},%
5131     sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
5132     text={\the\glsshorttok},%
5133     short={\the\glsshorttok},%
5134     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5135     long={\the\glslongtok},%
5136     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5137     \GenericAcronymFields,%
5138     \the\glskeylisttok
5139   }%
5140 }%
5141 \do@newglossaryentry
5142 }%

```

Make sure that \acrfull etc reflects the new style:

```

5143 \renewcommand*{\acrfullfmt}[3]{%
5144   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
5145 \renewcommand*{\Acrfullfmt}[3]{%
5146   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
5147 \renewcommand*{\ACRfullfmt}[3]{%
5148   \glslink[##1]{##2}{%
5149     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
5150 \renewcommand*{\acrfullplfmt}[3]{%
5151   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
5152 \renewcommand*{\Acrfullplfmt}[3]{%
5153   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
5154 \renewcommand*{\ACRfullplfmt}[3]{%
5155   \glslink[##1]{##2}{%
5156     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%

```

Make sure that \glsentryfull etc reflects the new style:

```

5157 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
5158 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
5159 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
5160 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
5161 }

```

`\GenericAcronymFields` Fields used by `\SetGenericNewAcronym` that can be changed by the acronym style.

```

5162 \newcommand*{\GenericAcronymFields}{description={\the\glslongtok}}

```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```

5163 \newcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{#1}}}

```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```
5164 \newcommand*{\acronymsort}[2]{#1}
```

`\setacronymstyle` `\setacronymstyle{<style name>}`

```
5165 \newcommand*{\setacronymstyle}[1]{%
5166   \ifcsundef{@glsacr@dispstyle@#1}%
5167   {%
5168     \PackageError{glossaries}{Undefined acronym style ‘#1’}{}%
5169   }%
5170   {%
5171     \ifdefempty{\@glsacronymlists}%
5172     {%
5173       \DeclareAcronymList{<acronym type>%
5174     }%
5175     {}%
5176     \SetGenericNewAcronym
5177     \GlsUseAcrStyleDefs{#1}%
5178     \@for\@gls@type:=\@glsacronymlists\do{%
5179       \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDispStyle{#1}}%
5180     }%
5181   }%
5182 }
```

`\newacronymstyle` `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called <style name>.

```
5183 \newcommand*{\newacronymstyle}[3]{%
5184   \ifcsdef{@glsacr@dispstyle@#1}%
5185   {%
5186     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5187   }%
5188   {%
5189     \csdef{@glsacr@dispstyle@#1}{#2}%
5190     \csdef{@glsacr@styledefs@#1}{#3}%
5191   }%
5192 }
```

`\renewacronymstyle` Redefines the given acronym style.

```
5193 \newcommand*{\renewacronymstyle}[3]{%
5194   \ifcsdef{@glsacr@dispstyle@#1}%
5195   {%
5196     \csdef{@glsacr@dispstyle@#1}{#2}%
5197     \csdef{@glsacr@styledefs@#1}{#3}%
5198   }%
```

```

5199  {%
5200    \PackageError{glossaries}{Acronym style ‘#1’ doesn’t exist}{}%
5201  }%
5202 }

```

seAcrEntryDispStyle

```

5203 \newcommand*{\GlsUseAcrEntryDispStyle}[1]{\csuse{@glsacr@dispstyle@#1}}

```

\GlsUseAcrStyleDefs

```

5204 \newcommand*{\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}

```

Predefined acronym styles:

long-short *⟨long⟩* (*⟨short⟩*) acronym style.

```

5205 \newacronymstyle{long-short}%
5206 {%

```

Check for long form in case this is a mixed glossary.

```

5207   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5208 }%
5209 {%
5210   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5211   \renewcommand*{\genacrfullformat}[2]{%
5212     \glsentrylong{##1}##2\space
5213     (\protect\firstacronymfont{\glsentryshort{##1}})%
5214   }%
5215   \renewcommand*{\Genacrfullformat}[2]{%
5216     \Glsentrylong{##1}##2\space
5217     (\protect\firstacronymfont{\glsentryshort{##1}})%
5218   }%
5219   \renewcommand*{\genplacrfullformat}[2]{%
5220     \glsentrylongpl{##1}##2\space
5221     (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5222   }%
5223   \renewcommand*{\Genplacrfullformat}[2]{%
5224     \Glsentrylongpl{##1}##2\space
5225     (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5226   }%
5227   \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
5228   \renewcommand*{\acronymsort}[2]{##1}%
5229   \renewcommand*{\acronymfont}[1]{##1}%
5230   \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5231   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5232 }

```

short-long *⟨short⟩* (*⟨long⟩*) acronym style.

```

5233 \newacronymstyle{short-long}%
5234 {%

```

Check for long form in case this is a mixed glossary.

```

5235 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5236 }%
5237 {%
5238 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5239 \renewcommand*{\genacrfullformat}[2]{%
5240 \protect\firstacronymfont{\glsenentryshort{##1}}##2\space
5241 (\glsenentrylong{##1})%
5242 }%
5243 \renewcommand*{\Genacrfullformat}[2]{%
5244 \protect\firstacronymfont{\Glsenentryshort{##1}}##2\space
5245 (\glsenentrylong{##1})%
5246 }%
5247 \renewcommand*{\genplacrfullformat}[2]{%
5248 \protect\firstacronymfont{\glsenentryshortpl{##1}}##2\space
5249 (\glsenentrylongpl{##1})%
5250 }%
5251 \renewcommand*{\Genplacrfullformat}[2]{%
5252 \protect\firstacronymfont{\Glsenentryshortpl{##1}}##2\space
5253 (\glsenentrylongpl{##1})%
5254 }%
5255 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsenentryshort{##1}}}%
5256 \renewcommand*{\acronymsort}[2]{##1}%
5257 \renewcommand*{\acronymfont}[1]{##1}%
5258 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5259 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5260 }

```

long-sc-short <long> (\textsc{<short>}) acronym style.

```

5261 \newacronymstyle{long-sc-short}%
5262 {%
5263 \GlsUseAcrEntryDisplayStyle{long-short}%
5264 }%
5265 {%
5266 \GlsUseAcrStyleDefs{long-short}%
5267 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5268 \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5269 }

```

long-sm-short <long> (\textsmaller{<short>}) acronym style.

```

5270 \newacronymstyle{long-sm-short}%
5271 {%
5272 \GlsUseAcrEntryDisplayStyle{long-short}%
5273 }%
5274 {%
5275 \GlsUseAcrStyleDefs{long-short}%
5276 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5277 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5278 }

```

sc-short-long *<short>* (\textsc{<long>}) acronym style.

```
5279 \newacronymstyle{sc-short-long}%
5280 {%
5281   \GlsUseAcrEntryDisplayStyle{short-long}%
5282 }%
5283 {%
5284   \GlsUseAcrStyleDefs{short-long}%
5285   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5286   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5287 }
```

sm-short-long *<short>* (\textsmaller{<long>}) acronym style.

```
5288 \newacronymstyle{sm-short-long}%
5289 {%
5290   \GlsUseAcrEntryDisplayStyle{short-long}%
5291 }%
5292 {%
5293   \GlsUseAcrStyleDefs{short-long}%
5294   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5295   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5296 }
```

long-short-desc *<long>* ({<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5297 \newacronymstyle{long-short-desc}%
5298 {%
5299   \GlsUseAcrEntryDisplayStyle{long-short}%
5300 }%
5301 {%
5302   \GlsUseAcrStyleDefs{long-short}%
5303   \renewcommand*{\GenericAcronymFields}{}%
5304   \renewcommand*{\acronymsort}[2]{##2}%
5305   \renewcommand*{\acronymentry}[1]{%
5306     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5307 }
```

long-sc-short-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5308 \newacronymstyle{long-sc-short-desc}%
5309 {%
5310   \GlsUseAcrEntryDisplayStyle{long-sc-short}%
5311 }%
5312 {%
5313   \GlsUseAcrStyleDefs{long-sc-short}%
5314   \renewcommand*{\GenericAcronymFields}{}%
5315   \renewcommand*{\acronymsort}[2]{##2}%
5316   \renewcommand*{\acronymentry}[1]{%
5317     \glsentrylong{##1}\space (\textsc{\acronymfont{\glsentryshort{##1}})}%
5318 }
```


long-sm-short-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5319 \newacronymstyle{long-sm-short-desc}%
5320 {%
5321   \GlsUseAcrEntryDispStyle{long-sm-short}%
5322 }%
5323 {%
5324   \GlsUseAcrStyleDefs{long-sm-short}%
5325   \renewcommand*{\GenericAcronymFields}{}%
5326   \renewcommand*{\acronymsort}[2]{##2}%
5327   \renewcommand*{\acronymentry}[1]{%
5328     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5329 }
```

short-long-desc *<short>* ({<long>}) acronym style that has an accompanying description (which the user needs to supply).

```
5330 \newacronymstyle{short-long-desc}%
5331 {%
5332   \GlsUseAcrEntryDispStyle{short-long}%
5333 }%
5334 {%
5335   \GlsUseAcrStyleDefs{short-long}%
5336   \renewcommand*{\GenericAcronymFields}{}%
5337   \renewcommand*{\acronymsort}[2]{##2}%
5338   \renewcommand*{\acronymentry}[1]{%
5339     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5340 }
```

sc-short-long-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5341 \newacronymstyle{sc-short-long-desc}%
5342 {%
5343   \GlsUseAcrEntryDispStyle{sc-short-long}%
5344 }%
5345 {%
5346   \GlsUseAcrStyleDefs{sc-short-long}%
5347   \renewcommand*{\GenericAcronymFields}{}%
5348   \renewcommand*{\acronymsort}[2]{##2}%
5349   \renewcommand*{\acronymentry}[1]{%
5350     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5351 }
```

sm-short-long-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5352 \newacronymstyle{sm-short-long-desc}%
5353 {%
5354   \GlsUseAcrEntryDispStyle{sm-short-long}%
5355 }
```

```

5356 {%
5357   \GlsUseAcrStyleDefs{sm-short-long}%
5358   \renewcommand*{\GenericAcronymFields}{}%
5359   \renewcommand*{\acronymsort}[2]{##2}%
5360   \renewcommand*{\acronymentry}[1]{%
5361     \glstrylong{##1}\space (\acronymfont{\glstryshort{##1}})}%
5362 }

```

dua *<long>* only acronym style.

```

5363 \newacronymstyle{dua}%
5364 {%

```

Check for long form in case this is a mixed glossary.

```

5365   \ifdefempty\glscustomtext
5366   {%
5367     \ifglshaslong{\glslabel}%
5368     {%
5369       \glssifplural
5370       {%

```

Plural form:

```

5371       \glscapscase
5372       {%

```

Plural form, don't adjust case:

```

5373       \glstrylongpl{\glslabel}\glssinsert
5374       }%
5375       {%

```

Plural form, make first letter upper case:

```

5376       \Glsentrylongpl{\glslabel}\glssinsert
5377       }%
5378       {%

```

Plural form, all caps:

```

5379       \mfirstucMakeUppercase
5380       {\glstrylongpl{\glslabel}\glssinsert}%
5381       }%
5382       }%
5383       {%

```

Singular form

```

5384       \glscapscase
5385       {%

```

Singular form, don't adjust case:

```

5386       \glstrylong{\glslabel}\glssinsert
5387       }%
5388       {%

```

Subsequent singular form, make first letter upper case:

```

5389       \Glsentrylong{\glslabel}\glssinsert

```

5390 }%
5391 {%

Subsequent singular form, all caps:

5392 \mfirstucMakeUppercase
5393 {\glsentrylong{\glslabel}\glsinsert}%
5394 }%
5395 }%
5396 }%
5397 {%

Not an acronym:

5398 \glsgenentryfmt
5399 }%
5400 }%
5401 {\glscustomtext\glsinsert}%
5402 }%
5403 {%
5404 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

5405 \renewcommand*{\acrfullfmt}[3]{%
5406 \glslink[##1]{##2}{\glsentrylong{##2}##3\space
5407 (\acronymfont{\glsentryshort{##2}})}}%
5408 \renewcommand*{\Acrfullfmt}[3]{%
5409 \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
5410 (\acronymfont{\glsentryshort{##2}})}}%
5411 \renewcommand*{\ACRfullfmt}[3]{%
5412 \glslink[##1]{##2}{%
5413 \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
5414 (\acronymfont{\glsentryshort{##2}})}}}%

5415 \renewcommand*{\acrfullplfmt}[3]{%
5416 \glslink[##1]{##2}{\glsentrylongpl{##2}##3\space
5417 (\acronymfont{\glsentryshortpl{##2}})}}%

5418 \renewcommand*{\Acrfullplfmt}[3]{%
5419 \glslink[##1]{##2}{\Glsentrylongpl{##2}##3\space
5420 (\acronymfont{\glsentryshortpl{##2}})}}%
5421 \renewcommand*{\ACRfullplfmt}[3]{%
5422 \glslink[##1]{##2}{%
5423 \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
5424 (\acronymfont{\glsentryshortpl{##2}})}}}%
5425 \renewcommand*{\glsentryfull}[1]{%
5426 \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
5427 }%
5428 \renewcommand*{\Glsentryfull}[1]{%
5429 \Glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
5430 }%
5431 \renewcommand*{\glsentryfullpl}[1]{%
5432 \glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
5433 }%

```

5434 \renewcommand*\Glsentryfullpl}[1]{%
5435   \Glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
5436 }%
5437 \renewcommand*\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
5438 \renewcommand*\acronymsort}[2]{##1}%
5439 \renewcommand*\acronymfont}[1]{##1}%
5440 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
5441 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

5442 \newacronymstyle{dua-desc}%
5443 {%
5444   \GlsUseAcrEntryDispStyle{dua}%
5445 }%
5446 {%
5447   \GlsUseAcrStyleDefs{dua}%
5448   \renewcommand*\GenericAcronymFields{}%

5449   \renewcommand*\acronymentry}[1]{\acronymfont{\glsentrylong{##1}}}%
5450   \renewcommand*\acronymsort}[2]{##2}%
5451 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

5452 \newacronymstyle{footnote}%
5453 {%

```

Check for long form in case this is a mixed glossary.

```

5454 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5455 }%
5456 {%
5457   \renewcommand*\GenericAcronymFields{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

5458 \glshyperfirstfalse
5459 \renewcommand*\genacrfullformat}[2]{%
5460   \protect\firstacronymfont{\glsentryshort{##1}}##2%
5461   \protect\footnote{\glsentrylong{##1}}%
5462 }%
5463 \renewcommand*\Genacrfullformat}[2]{%
5464   \firstacronymfont{\Glsentryshort{##1}}##2%
5465   \protect\footnote{\glsentrylong{##1}}%
5466 }%
5467 \renewcommand*\genplacrfullformat}[2]{%
5468   \protect\firstacronymfont{\glsentryshortpl{##1}}##2%
5469   \protect\footnote{\glsentrylongpl{##1}}%
5470 }%
5471 \renewcommand*\Genplacrfullformat}[2]{%
5472   \protect\firstacronymfont{\Glsentryshortpl{##1}}##2%
5473   \protect\footnote{\glsentrylongpl{##1}}%
5474 }%

```

```

5475 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
5476 \renewcommand*{\acronymsort}[2]{##1}%
5477 \renewcommand*{\acronymfont}[1]{##1}%
5478 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%

```

Don't use footnotes for \acrfull:

```

5479 \renewcommand*{\acrfullfmt}[3]{%
5480   \glslink[##1]{##2}{\acronymfont{\glentryshort{##2}}##3\space
5481     (\glentrylong{##2})}%
5482 \renewcommand*{\Acrfullfmt}[3]{%
5483   \glslink[##1]{##2}{\acronymfont{\Glentryshort{##2}}##3\space
5484     (\glentrylong{##2})}%
5485 \renewcommand*{\ACRfullfmt}[3]{%
5486   \glslink[##1]{##2}{%
5487     \mfirstucMakeUppercase{\acronymfont{\glentryshort{##2}}##3\space
5488       (\glentrylong{##2})}}}%
5489 \renewcommand*{\acrfullplfmt}[3]{%
5490   \glslink[##1]{##2}{\acronymfont{\glentryshortpl{##2}}##3\space
5491     (\glentrylongpl{##2})}%
5492 \renewcommand*{\Acrfullplfmt}[3]{%
5493   \glslink[##1]{##2}{\acronymfont{\Glentryshortpl{##2}}##3\space
5494     (\glentrylongpl{##2})}%
5495 \renewcommand*{\ACRfullplfmt}[3]{%
5496   \glslink[##1]{##2}{%
5497     \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{##2}}##3\space
5498       (\glentrylongpl{##2})}}}%

```

Similarly for \glentryfull etc:

```

5499 \renewcommand*{\glentryfull}[1]{%
5500   \acronymfont{\glentryshort{##1}}\space(\glentrylong{##1})}%
5501 \renewcommand*{\Glentryfull}[1]{%
5502   \acronymfont{\Glentryshort{##1}}\space(\glentrylong{##1})}%
5503 \renewcommand*{\glentryfullpl}[1]{%
5504   \acronymfont{\glentryshortpl{##1}}\space(\glentrylongpl{##1})}%
5505 \renewcommand*{\Glentryfullpl}[1]{%
5506   \acronymfont{\Glentryshortpl{##1}}\space(\glentrylongpl{##1})}%
5507 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

5508 \newacronymstyle{footnote-sc}%
5509 {%
5510   \GlsUseAcrEntryDispStyle{footnote}%
5511 }%
5512 {%
5513   \GlsUseAcrStyleDefs{footnote}%
5514   \renewcommand{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
5515   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5516   \renewcommand*{\acrpluralsuffix}{\glstextup{\glpluralsuffix}}%
5517 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```
5518 \newacronymstyle{footnote-sm}%
5519 {%
5520 \GlsUseAcrEntryDisplayStyle{footnote}%
5521 }%
5522 {%
5523 \GlsUseAcrStyleDefs{footnote}%
5524 \renewcommand{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
5525 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5526 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%
5527 }
```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
5528 \newacronymstyle{footnote-desc}%
5529 {%
5530 \GlsUseAcrEntryDisplayStyle{footnote}%
5531 }%
5532 {%
5533 \GlsUseAcrStyleDefs{footnote}%
5534 \renewcommand*{\GenericAcronymFields}{}%
5535 \renewcommand*{\acronymsort}[2]{##2}%
5536 \renewcommand*{\acronymentry}[1]{%
5537 \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5538 }
```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
5539 \newacronymstyle{footnote-sc-desc}%
5540 {%
5541 \GlsUseAcrEntryDisplayStyle{footnote-sc}%
5542 }%
5543 {%
5544 \GlsUseAcrStyleDefs{footnote-sc}%
5545 \renewcommand*{\GenericAcronymFields}{}%
5546 \renewcommand*{\acronymsort}[2]{##2}%
5547 \renewcommand*{\acronymentry}[1]{%
5548 \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5549 }
```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
5550 \newacronymstyle{footnote-sm-desc}%
5551 {%
5552 \GlsUseAcrEntryDisplayStyle{footnote-sm}%
5553 }%
5554 {%
5555 \GlsUseAcrStyleDefs{footnote-sm}%
5556 }
```

```

5556 \renewcommand*{\GenericAcronymFields}{}%
5557 \renewcommand*{\acronymsort}[2]{##2}%
5558 \renewcommand*{\acronymentry}[1]{%
5559     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5560 }

```

`\fineAcronymSynonyms`

```

5561 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

`\acs`

```

5562 \let\acs\acrshort

```

First letter uppercase short form

`\Acs`

```

5563 \let\Acs\Acrshort

```

Plural short form

`\acsp`

```

5564 \let\acsp\acrshortpl

```

First letter uppercase plural short form

`\Acsp`

```

5565 \let\Acsp\Acrshortpl

```

Long form

`\acl`

```

5566 \let\acl\aclong

```

Plural long form

`\aclp`

```

5567 \let\aclp\aclongpl

```

First letter upper case long form

`\Acl`

```

5568 \let\Acl\Aclong

```

First letter upper case plural long form

`\Aclp`

```

5569 \let\Aclp\Aclongpl

```

Full form

`\acf`

```

5570 \let\acf\acrfull

```

Plural full form

`\acfp`

```
5571 \let\acfp\acrfullpl
```

First letter upper case full form

`\Acf`

```
5572 \let\Acf\Acrfull
```

First letter upper case plural full form

`\Acfp`

```
5573 \let\Acfp\Acrfullpl
```

Standard form

`\ac`

```
5574 \let\ac\gls
```

First upper case standard form

`\Ac`

```
5575 \let\Ac\Gls
```

Standard plural form

`\acp`

```
5576 \let\acp\glspl
```

Standard first letter upper case plural form

`\Acp`

```
5577 \let\Acp\Glspl
```

```
5578 }
```

Define synonyms if required

```
5579 \ifglsacrshortcuts
```

```
5580 \DefineAcronymSynonyms
```

```
5581 \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.

```
5582 \newcommand*\SetDefaultAcronymDisplayStyle}[1]{%
```

```
5583 \defglsentryfmt[#1]{\glsentryfmt}%
```

```
5584 }
```


`\defaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```

5585 \newcommand*{\DefaultNewAcronymDef}{%
5586   \edef\@do@newglossaryentry{%
5587     \noexpand\newglossaryentry{\the\glslabeltok}%
5588     {%
5589       type=\acronymtype,%
5590       name={\the\glsshorttok},%
5591       sort={\the\glsshorttok},%
5592       text={\the\glsshorttok},%
5593       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
5594       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5595       firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
5596                   {\noexpand\expandonce\noexpand\@glo@shortpl}},%
5597       short={\the\glsshorttok},%
5598       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5599       long={\the\glslongtok},%
5600       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5601       description={\the\glslongtok},%
5602       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

Remaining options specified by the user:

```

5603     \the\glskeylisttok
5604   }%
5605 }%
5606 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5607 \let\@org@gls@assign@plural\gls@assign@plural
5608 \let\@org@gls@assign@descplural\gls@assign@descplural
5609 \def\gls@assign@firstpl##1##2{%
5610   \@@gls@expand@field{##1}{firstpl}{##2}%
5611 }%
5612 \def\gls@assign@plural##1##2{%
5613   \@@gls@expand@field{##1}{plural}{##2}%
5614 }%
5615 \def\gls@assign@descplural##1##2{%
5616   \@@gls@expand@field{##1}{descplural}{##2}%
5617 }%
5618 \@do@newglossaryentry
5619 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5620 \let\gls@assign@plural\@org@gls@assign@plural
5621 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5622 }

```

`\DefaultAcronymStyle` Set up the default acronym style:

```

5623 \newcommand*{\SetDefaultAcronymStyle}{%
    Set the display style:
5624   \@for\@gls@type:=\@glsacronymlists\do{%
5625     \SetDefaultAcronymDisplayStyle{\@gls@type}%

```

5626 }%

Set up the definition of \newacronym:

5627 \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).

```
5628 \ifx\@glsacronymlists\@empty
5629 \def\@glo@type{\acronymtype}%
5630 \setkeys{glossentry}{##1}%
5631 \DeclareAcronymList{\@glo@type}%
5632 \SetDefaultAcronymDisplayStyle{\@glo@type}%
5633 \fi
5634 \glskeylisttok{##1}%
5635 \glslabeltok{##2}%
5636 \glsshorttok{##3}%
5637 \gslongtok{##4}%
5638 \newacronymhook
5639 \DefaultNewAcronymDef
5640 }%
5641 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5642 }
```

\acrfootnote Used by the footnote acronym styles.

5643 \newcommand*{\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

\acrlinkfootnote

```
5644 \newcommand*{\acrlinkfootnote}[3]{%
5645 \footnote{\glslink[#1]{#2}{#3}}%
5646 }
```

\acrlinkfootnote

```
5647 \newcommand*{\acrlinkfootnote}[3]{%
5648 \footnote{#3}%
5649 }
```

AcronymDisplayStyle Sets the acronym display style for given glossary for the description and footnote combination.

```
5650 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
5651 \def\glsentryfmt[#1]{%
5652 \ifdefempty\glscustomtext
5653 {%
5654 \ifglsused{\glslabel}%
5655 {%
5656 \acronymfont{\glsentryfmt}%
5657 }%
5658 }%
```

```

5659     \firstacronymfont{\glsentryfmt}%
5660     \ifglshassymbol{\glslabel}%
5661     {%
5662         \expandafter\protect\expandafter\acrfootnote\expandafter
5663         {\@gls@link@opts}{\@gls@link@label}%
5664         {%
5665             \glsifplural
5666             {\glsentrysymbolplural{\glslabel}}%
5667             {\glsentrysymbol{\glslabel}}%
5668         }%
5669     }%
5670 }%
5671 }%
5672 {\glscustomtext\glsinsert}%
5673 }%
5674 }

```

otnoteNewAcronymDef

```

5675 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
5676     \edef\@do@newglossaryentry{%
5677         \noexpand\newglossaryentry{\the\glslabeltok}%
5678         {%
5679             type=\acronymtype,%
5680             name={\noexpand\acronymfont{\the\glsshorttok}},%
5681             sort={\the\glsshorttok},%
5682             first={\the\glsshorttok},%
5683             firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5684             text={\the\glsshorttok},%
5685             plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5686             short={\the\glsshorttok},%
5687             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5688             long={\the\glslongtok},%
5689             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5690             symbol={\the\glslongtok},%
5691             symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5692             \the\glskeylisttok
5693         }%
5694     }%
5695     \let\@org@gls@assign@firstpl\gls@assign@firstpl
5696     \let\@org@gls@assign@plural\gls@assign@plural
5697     \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5698     \def\gls@assign@firstpl##1##2{%
5699         \@@gls@expand@field{##1}{firstpl}{##2}%
5700     }%
5701     \def\gls@assign@plural##1##2{%
5702         \@@gls@expand@field{##1}{plural}{##2}%
5703     }%
5704     \def\gls@assign@symbolplural##1##2{%
5705         \@@gls@expand@field{##1}{symbolplural}{##2}%

```

```

5706 }%
5707 \do@newglossaryentry
5708 \let\gls@assign@plural\@org@gls@assign@plural
5709 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5710 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5711 }

```

footnoteAcronymStyle 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.

```

5712 \newcommand*\SetDescriptionFootnoteAcronymStyle{%
5713   \renewcommand{\newacronym}[4][\]{%
5714     \ifx\@glsacronymlists\@empty
5715       \def\@glo@type{\acronymtype}%
5716       \setkeys{glossentry}{##1}%
5717       \DeclareAcronymList{\@glo@type}%
5718       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
5719     \fi
5720     \glskeylisttok{##1}%
5721     \glslabeltok{##2}%
5722     \glsshorttok{##3}%
5723     \gslongtok{##4}%
5724     \newacronymhook
5725     \DescriptionFootnoteNewAcronymDef
5726   }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

5727 \@for\@gls@type:=\@glsacronymlists\do{%
5728   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
5729 }%

```

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.

```

5730 \ifglsacrsmallcaps
5731   \renewcommand*\acronymfont{[1]{\textsc{##1}}}%
5732   \renewcommand*\acrpluralsuffix{%
5733     \glstextup{\glspluralsuffix}}%
5734 \else
5735   \ifglsacrsmaller
5736     \renewcommand*\acronymfont{[1]{\textsmaller{##1}}}%
5737   \fi
5738 \fi

```

Check for package option clash

```

5739 \ifglsacrdua
5740   \PackageError{glossaries}{Option clash: 'footnote' and 'dua'}

```

```

5741     can't both be set}{}%
5742   \fi
5743 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

5744 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
5745   \def\glentryfmt[#1]{\glsgenentryfmt}%
5746 }

```

ionDUANewAcronymDef

```

5747 \newcommand*{\DescriptionDUANewAcronymDef}{%
5748   \edef\@do@newglossaryentry{%
5749     \noexpand\newglossaryentry{\the\glslabeltok}%
5750     {%
5751       type=\acronymtype,%
5752       name={\the\glslongtok},%
5753       sort={\the\glslongtok},%
5754       text={\the\glslongtok},%
5755       first={\the\glslongtok},%
5756       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
5757       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5758       short={\the\glsshorttok},%
5759       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5760       long={\the\glslongtok},%
5761       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5762       symbol={\the\glsshorttok},%
5763       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5764       \the\glskeylisttok
5765     }%
5766   }%
5767   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5768   \let\@org@gls@assign@plural\gls@assign@plural
5769   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5770   \def\gls@assign@firstpl##1##2{%
5771     \@@gls@expand@field{##1}{firstpl}{##2}%
5772   }%
5773   \def\gls@assign@plural##1##2{%
5774     \@@gls@expand@field{##1}{plural}{##2}%
5775   }%
5776   \def\gls@assign@symbolplural##1##2{%
5777     \@@gls@expand@field{##1}{symbolplural}{##2}%
5778   }%
5779   \@do@newglossaryentry
5780   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5781   \let\gls@assign@plural\@org@gls@assign@plural
5782   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5783 }

```

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.

```

5784 \newcommand*{\SetDescriptionDUAAcronymStyle}{%
5785   \ifglsmallcaps
5786     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
5787       can't both be set}{}%
5788   \else
5789     \ifglsmaller
5790       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
5791         can't both be set}{}%
5792     \fi
5793   \fi
5794   \renewcommand{\newacronym}[4][[]]{%
5795     \ifx\@glsacronymlists\@empty
5796       \def\@glo@type{\acronymtype}%
5797       \setkeys{glossentry}{##1}%
5798       \DeclareAcronymList{\@glo@type}%
5799       \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
5800     \fi
5801     \glskeylisttok{##1}%
5802     \glslabeltok{##2}%
5803     \glsshorttok{##3}%
5804     \gslongtok{##4}%
5805     \newacronymhook
5806     \DescriptionDUANewAcronymDef
5807   }%

```

Set display.

```

5808   \@for\@gls@type:=\@glsacronymlists\do{%
5809     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
5810   }%
5811 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

5812 \newcommand*{\SetDescriptionAcronymDisplayStyle}[1]{%
5813   \defglentryfmt[#1]{%
5814     \ifdefempty\glscustomtext
5815     {%
5816       \ifglused{\glslabel}%
5817     {%

```

Move the inserted text outside of \acronymfont

```

5818       \let\gls@org@insert\glsinsert
5819       \let\glsinsert\@empty
5820       \acronymfont{\glsgenentryfmt}\gls@org@insert
5821     }%

```

```

5822     {%
5823         \glsgenentryfmt
5824         \ifglshassymbol{\glslabel}%
5825         {%
5826             \glsifplural
5827             {%
5828                 \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
5829             }%
5830             {%
5831                 \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
5832             }%
5833             \space(\protect\firstacronymfont
5834             {\glscapscase
5835              {\@glo@symbol}
5836              {\@glo@symbol}
5837              {\mfirstucMakeUppercase{\@glo@symbol}}})%
5838         }%
5839     }%
5840 }%
5841 }%
5842 {\glscustomtext\glsinsert}%
5843 }%
5844 }

```

ptionNewAcronymDef

```

5845 \newcommand*{\DescriptionNewAcronymDef}{%
5846     \edef\@do@newglossaryentry{%
5847         \noexpand\newglossaryentry{\the\glslabeltok}%
5848         {%
5849             type=\acronymtype,%
5850             name={\noexpand
5851                 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
5852             sort={\the\glsshorttok},%
5853             first={\the\glslongtok},%
5854             firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5855             text={\the\glsshorttok},%
5856             plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5857             short={\the\glsshorttok},%
5858             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5859             long={\the\glslongtok},%
5860             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5861             symbol={\noexpand\@glo@text},%
5862             symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5863             \the\glskeylisttok}%
5864     }%
5865     \let\@org@gls@assign@firstpl\gls@assign@firstpl
5866     \let\@org@gls@assign@plural\gls@assign@plural
5867     \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5868     \def\gls@assign@firstpl##1##2{%

```

```

5869 \@@gls@expand@field{##1}{firstpl}{##2}%
5870 }%
5871 \def\gls@assign@plural##1##2{%
5872 \@@gls@expand@field{##1}{plural}{##2}%
5873 }%
5874 \def\gls@assign@symbolplural##1##2{%
5875 \@@gls@expand@field{##1}{symbolplural}{##2}%
5876 }%
5877 \do@newglossaryentry
5878 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5879 \let\gls@assign@plural\@org@gls@assign@plural
5880 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5881 }

```

DescriptionAcronymStyle 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.

```

5882 \newcommand*{\SetDescriptionAcronymStyle}{%
5883 \renewcommand{\newacronym}[4][\]{%
5884 \ifx\@glsacronymlists\@empty
5885 \def\@glo@type{\acronymtype}%
5886 \setkeys{glossentry}{##1}%
5887 \DeclareAcronymList{\@glo@type}%
5888 \SetDescriptionAcronymDisplayStyle{\@glo@type}%
5889 \fi
5890 \glskeylisttok{##1}%
5891 \glslabeltok{##2}%
5892 \glsshorttok{##3}%
5893 \glslongtok{##4}%
5894 \newacronymhook
5895 \DescriptionNewAcronymDef
5896 }%

```

Set display.

```

5897 \@for\@gls@type:=\@glsacronymlists\do{%
5898 \SetDescriptionAcronymDisplayStyle{\@gls@type}%
5899 }%

```

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.

```

5900 \ifglsacrsmallcaps
5901 \renewcommand{\acronymfont}[1]{\textsc{##1}}
5902 \renewcommand*{\acrpluralsuffix}{%
5903 \glstextup{\glspluralsuffix}}%
5904 \else
5905 \ifglsacrsmaller
5906 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%

```



```

5907    \fi
5908    \fi
5909 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

5910 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
5911    \def\glsentryfmt[#1]{%

5912        \ifdefempty\glscustomtext
5913        {%

            Move the inserted text outside of \acronymfont

5914            \let\gls@org@insert\glsinsert
5915            \let\glsinsert\@empty
5916            \ifglssused{\glslabel}%
5917            {%
5918                \acronymfont{\glsentryfmt}\gls@org@insert
5919            }%
5920            {%
5921                \firstacronymfont{\glsentryfmt}\gls@org@insert
5922                \ifglshaslong{\glslabel}%
5923                {%
5924                    \expandafter\protect\expandafter\acrfootnote\expandafter
5925                    {\@gls@link@opts}{\@gls@link@label}%
5926                    {%
5927                        \glsifplural
5928                        {\glsentrylongpl{\glslabel}}%
5929                        {\glsentrylong{\glslabel}}%
5930                    }%
5931                }%

5932            }%
5933        }%
5934    }%
5935    {\glscustomtext\glsinsert}%
5936 }%
5937 }

```

FootnoteNewAcronymDef

```

5938 \newcommand*{\FootnoteNewAcronymDef}{%
5939    \edef\@do@newglossaryentry{%
5940        \noexpand\newglossaryentry{\the\glslabeltok}%
5941        {%
5942            type=\acronymtype,%
5943            name={\noexpand\acronymfont{\the\glsshorttok}},%
5944            sort={\the\glsshorttok},%
5945            text={\the\glsshorttok},%
5946            plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5947            first={\the\glsshorttok},%

```

```

5948     firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5949     short={\the\glsshorttok},%
5950     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5951     long={\the\glslongtok},%
5952     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5953     description={\the\glslongtok},%
5954     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5955     \the\glskeylisttok
5956 }%
5957 }%
5958 \let\@org@gls@assign@plural\gls@assign@plural
5959 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5960 \let\@org@gls@assign@descplural\gls@assign@descplural
5961 \def\gls@assign@firstpl##1##2{%
5962   \@@gls@expand@field{##1}{firstpl}{##2}%
5963 }%
5964 \def\gls@assign@plural##1##2{%
5965   \@@gls@expand@field{##1}{plural}{##2}%
5966 }%
5967 \def\gls@assign@descplural##1##2{%
5968   \@@gls@expand@field{##1}{descplural}{##2}%
5969 }%
5970 \do@newglossaryentry
5971 \let\gls@assign@plural\@org@gls@assign@plural
5972 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5973 \let\gls@assign@descplural\@org@gls@assign@descplural
5974 }

```

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.

```

5975 \newcommand*\SetFootnoteAcronymStyle{%
5976   \renewcommand{\newacronym}[4][{}]{%
5977     \ifx\@glsacronymlists\@empty
5978       \def\@glo@type{\acronymtype}%
5979       \setkeys{glossentry}{##1}%
5980       \DeclareAcronymList{\@glo@type}%
5981       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
5982     \fi
5983     \glskeylisttok{##1}%
5984     \glslabeltok{##2}%
5985     \glsshorttok{##3}%
5986     \glslongtok{##4}%
5987     \newacronymhook
5988     \FootnoteNewAcronymDef
5989   }%

```

Set display

```

5990   \@for\@gls@type:=\@glsacronymlists\do{%

```

```

5991 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
5992 }%

```

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.

```

5993 \ifglsacrsmallcaps
5994 \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
5995 \renewcommand*{\acrpluralsuffix}{%
5996 \glstextup{\glspluralsuffix}}%
5997 \else
5998 \ifglsacrsmaller
5999 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6000 \fi
6001 \fi

```

Check for option clash

```

6002 \ifglsacrdua
6003 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6004 can’t both be set}}{}%
6005 \fi
6006 }%

```

\glsdoparenifnotempty 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.

```

6007 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
6008 \protected@edef\gls@tmp{#1}%
6009 \ifdefempty\gls@tmp
6010 {}%
6011 {%
6012 \ifx\gls@tmp\@gls@default@value
6013 \else
6014 \space (#2{#1})%
6015 \fi
6016 }%
6017 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

6018 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
6019 \defglsentryfmt[#1]{%
6020 \ifdefempty\glscustomtext
6021 {%
6022 \let\gls@org@insert\glsinsert
6023 \let\glsinsert\@empty
6024 \ifglsused{\glslabel}%

```

```

6025     {%
6026     \acronymfont{\glsgenentryfmt}\gls@org@insert
6027     }%
6028     {%
6029     \glsgenentryfmt
6030     \ifglshassymbol{\glslabel}%
6031     {%
6032     \glsifplural
6033     {%
6034     \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6035     }%
6036     {%
6037     \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6038     }%
6039     \space
6040     (\glscapscase
6041     {\firstacronymfont{\@glo@symbol}}%
6042     {\firstacronymfont{\@glo@symbol}}%
6043     {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6044     }%
6045     {}%
6046     }%
6047     }%
6048     {\glscustomtext\glsinsert}%
6049     }%
6050 }

```

\SmallNewAcronymDef

```

6051 \newcommand*{\SmallNewAcronymDef}{%
6052   \edef\@do@newglossaryentry{%
6053     \noexpand\newglossaryentry{\the\glslabeltok}%
6054     {%
6055       type=\acronymtype,%
6056       name={\noexpand\acronymfont{\the\glsshorttok}},%
6057       sort={\the\glsshorttok},%
6058       text={\the\glsshorttok},%

```

Default to the short plural.

```

6059     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6060     first={\the\glslongtok},%

```

Default to the long plural.

```

6061     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6062     short={\the\glsshorttok},%
6063     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6064     long={\the\glslongtok},%
6065     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6066     description={\noexpand\@glo@first},%
6067     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6068     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6069     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6070     \the\glskeylisttok
6071   }%
6072 }%
6073 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6074 \let\@org@gls@assign@plural\gls@assign@plural
6075 \let\@org@gls@assign@descplural\gls@assign@descplural
6076 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6077 \def\gls@assign@firstpl##1##2{%
6078   \@@gls@expand@field{##1}{firstpl}{##2}%
6079 }%
6080 \def\gls@assign@plural##1##2{%
6081   \@@gls@expand@field{##1}{plural}{##2}%
6082 }%
6083 \def\gls@assign@descplural##1##2{%
6084   \@@gls@expand@field{##1}{descplural}{##2}%
6085 }%
6086 \def\gls@assign@symbolplural##1##2{%
6087   \@@gls@expand@field{##1}{symbolplural}{##2}%
6088 }%
6089 \@do@newglossaryentry
6090 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6091 \let\gls@assign@plural\@org@gls@assign@plural
6092 \let\gls@assign@descplural\@org@gls@assign@descplural
6093 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6094 }

```

`\setSmallAcronymStyle` 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.

```

6095 \newcommand*{\SetSmallAcronymStyle}{%
6096   \renewcommand{\newacronym}[4][]{%
6097     \ifx\@glsacronymlists\@empty
6098       \def\@glo@type{\acronymtype}%
6099       \setkeys{glossentry}{##1}%
6100       \DeclareAcronymList{\@glo@type}%
6101       \SetSmallAcronymDisplayStyle{\@glo@type}%
6102     \fi
6103     \glskeylisttok{##1}%
6104     \glslabeltok{##2}%
6105     \glsshorttok{##3}%
6106     \glslongtok{##4}%
6107     \newacronymhook
6108     \SmallNewAcronymDef
6109   }%

```

Change the display since first only contains long form.

```

6110   \@for\@gls@type:=\@glsacronymlists\do{%
6111     \SetSmallAcronymDisplayStyle{\@gls@type}%

```

6112 }%

Redefine \acronymfont if small caps required. The plural suffix is set in an up-right font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```
6113 \ifglsmallcaps
6114   \renewcommand*{\acronymfont}[1]{\textsc{##1}}
6115   \renewcommand*{\acrpluralsuffix}{%
6116     \glstextup{\glspluralsuffix}}%
6117 \else
6118   \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
6119 \fi
```

check for option clash

```
6120 \ifglsmallcaps
6121   \ifglsmallcaps
6122     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6123       can't both be set}{}%
6124   \else
6125     \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6126       can't both be set}{}%
6127 \fi
6128 \fi
6129 }%
```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```
6130 \newcommand*{\SetDUADisplayStyle}[1]{%
6131   \defglentryfmt[#1]{\glsgenentryfmt}%
6132 }
```

\DUANewAcronymDef

```
6133 \newcommand*{\DUANewAcronymDef}{%
6134   \edef\@do@newglossaryentry{%
6135     \noexpand\newglossaryentry{\the\glslabeltok}%
6136     {%
6137       type=\acronymtype,%
6138       name={\the\glsshorttok},%
6139       text={\the\glslongtok},%
6140       first={\the\glslongtok},%
6141       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6142       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6143       short={\the\glsshorttok},%
6144       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6145       long={\the\glslongtok},%
6146       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6147       description={\the\glslongtok},%
6148       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6149       symbol={\the\glsshorttok},%
6150       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6151       \the\glskeylisttok
```

```

6152 }%
6153 }%
6154 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
6155 \let\@org@gl@s@assign@plural\gl@s@assign@plural
6156 \let\@org@gl@s@assign@symbolplural\gl@s@assign@symbolplural
6157 \let\@org@gl@s@assign@descplural\gl@s@assign@descplural
6158 \def\gl@s@assign@firstpl##1##2{%
6159   \@@gl@s@expand@field{##1}{firstpl}{##2}%
6160 }%
6161 \def\gl@s@assign@plural##1##2{%
6162   \@@gl@s@expand@field{##1}{plural}{##2}%
6163 }%
6164 \def\gl@s@assign@symbolplural##1##2{%
6165   \@@gl@s@expand@field{##1}{symbolplural}{##2}%
6166 }%
6167 \def\gl@s@assign@descplural##1##2{%
6168   \@@gl@s@expand@field{##1}{descplural}{##2}%
6169 }%
6170 \do@newglossaryentry
6171 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
6172 \let\gl@s@assign@plural\@org@gl@s@assign@plural
6173 \let\gl@s@assign@symbolplural\@org@gl@s@assign@symbolplural
6174 \let\gl@s@assign@descplural\@org@gl@s@assign@descplural
6175 }

```

`\SetDUASStyle` Always expand acronyms.

```

6176 \newcommand*\SetDUASStyle{%
6177   \renewcommand{\newacronym}[4][\]{%
6178     \ifx\@gl@s@acronymlists\@empty
6179       \def\@glo@type{\acronymtype}%
6180       \setkeys{glossentry}{##1}%
6181       \DeclareAcronymList{\@glo@type}%
6182       \SetDUADisplayStyle{\@glo@type}%
6183     \fi
6184     \gl@s@keylisttok{##1}%
6185     \gl@s@labeltok{##2}%
6186     \gl@s@shorttok{##3}%
6187     \gl@s@longtok{##4}%
6188     \newacronymhook
6189     \DUANewAcronymDef
6190   }%
6191   \Set the display
6192   \@for\@gl@s@type:=\@gl@s@acronymlists\do{%
6193     \SetDUADisplayStyle{\@gl@s@type}%
6194   }%

```

`\SetAcronymStyle`

```

6195 \newcommand*{\SetAcronymStyle}{%
6196   \SetDefaultAcronymStyle
6197   \ifglssacrdescription
6198     \ifglssacrfootnote
6199       \SetDescriptionFootnoteAcronymStyle
6200     \else
6201       \ifglssacrdua
6202         \SetDescriptionDUAAcronymStyle
6203       \else
6204         \SetDescriptionAcronymStyle
6205       \fi
6206     \fi
6207   \else
6208     \ifglssacrfootnote
6209       \SetFootnoteAcronymStyle
6210     \else
6211       \ifthenelse{\boolean{glssacrsmalls} \OR
6212         \boolean{glssacrsmalls}}{%
6213         {%
6214           \SetSmallAcronymStyle
6215         }%
6216         {%
6217           \ifglssacrdua
6218             \SetDUASStyle
6219           \fi
6220         }%
6221       \fi
6222     \fi
6223 }

```

Set the acronym style according to the package options

```
6224 \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.

```

6225 \newcommand*{\SetCustomDisplayStyle}[1]{%
6226   \defglssentryfmt[#1]{\glsgenentryfmt}%
6227 }

```

`\CustomAcronymFields`

```

6228 \newcommand*{\CustomAcronymFields}{%
6229   name={\the\glssshorttok},%
6230   description={\the\glslongtok},%
6231   first={\noexpand\acrfullformat{\the\glslongtok}{\the\glssshorttok}},%

```



```

6232 firstplural={\noexpand\acrfullformat
6233   {\noexpand\glsentrylongpl{\the\glslabeltok}}}%
6234   {\noexpand\glsentryshortpl{\the\glslabeltok}}},%

6235 text={\the\glsshorttok},%
6236 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
6237 }

```

CustomNewAcronymDef

```

6238 \newcommand*{\CustomNewAcronymDef}{%
6239   \protected@edef\@do@newglossaryentry{%
6240     \noexpand\newglossaryentry{\the\glslabeltok}%
6241     {%
6242       type=\acronymtype,%
6243       short={\the\glsshorttok},%
6244       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6245       long={\the\glslongtok},%
6246       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6247       user1={\the\glsshorttok},%
6248       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
6249       user3={\the\glslongtok},%
6250       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
6251       \CustomAcronymFields,%
6252       \the\glskeylisttok
6253     }%
6254   }%
6255   \@do@newglossaryentry
6256 }

```

\SetCustomStyle

```

6257 \newcommand*{\SetCustomStyle}{%
6258   \renewcommand{\newacronym}[4][\]{%
6259     \ifx\@glsacronymlists\@empty
6260       \def\@glo@type{\acronymtype}%
6261       \setkeys{glossentry}{##1}%
6262       \DeclareAcronymList{\@glo@type}%
6263       \SetCustomDisplayStyle{\@glo@type}%
6264     \fi
6265     \glskeylisttok{##1}%
6266     \glslabeltok{##2}%
6267     \glsshorttok{##3}%
6268     \glslongtok{##4}%
6269     \newacronymhook
6270     \CustomNewAcronymDef
6271   }%

   Set the display
6272   \@for\@gls@type:=\@glsacronymlists\do{%
6273     \SetCustomDisplayStyle{\@gls@type}%
6274   }%

```

6275 }

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.

6276 \RequirePackage{glossary-hypernav}

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

6277 \@gls@loadlist

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

6278 \@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.

6279 \@gls@loadsuper

The tree-like styles. These are not loaded if the `notree` package option is used.

6280 \@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.

6281 \ifx\@glossary@default@style\relax

6282 \else

6283 \setglossarystyle{\@glossary@default@style}

6284 \fi

1.19 Debugging Commands

`\showgloparent` `\showgloparent{<label>}`

6285 \newcommand*{\showgloparent}[1]{%

6286 \expandafter\show\csname glo@\glsdetoklabel{#1}@parent\endcsname

6287 }

`\showglolevel` `\showglolevel{<label>}`

6288 \newcommand*{\showglolevel}[1]{%

6289 \expandafter\show\csname glo@\glsdetoklabel{#1}@level\endcsname

6290 }

`\showglotext` `\showglotext{<label>}`

```
6291 \newcommand*{\showglotext}[1]{%
6292   \expandafter\show\csname glo@\glsdetoklabel{#1}@text\endcsname
6293 }
```

`\showgloplural` `\showgloplural{<label>}`

```
6294 \newcommand*{\showgloplural}[1]{%
6295   \expandafter\show\csname glo@\glsdetoklabel{#1}@plural\endcsname
6296 }
```

`\showglofirst` `\showglofirst{<label>}`

```
6297 \newcommand*{\showglofirst}[1]{%
6298   \expandafter\show\csname glo@\glsdetoklabel{#1}@first\endcsname
6299 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
6300 \newcommand*{\showglofirstpl}[1]{%
6301   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpl\endcsname
6302 }
```

`\showglotype` `\showglotype{<label>}`

```
6303 \newcommand*{\showglotype}[1]{%
6304   \expandafter\show\csname glo@\glsdetoklabel{#1}@type\endcsname
6305 }
```

`\showglocounter` `\showglocounter{<label>}`

```
6306 \newcommand*{\showglocounter}[1]{%
6307   \expandafter\show\csname glo@\glsdetoklabel{#1}@counter\endcsname
6308 }
```

\showglouserii \showglouserii{\label{}}

```
6309 \newcommand*{\showglouserii}[1]{%
6310   \expandafter\show\csname glo@\glsdetoklabel{#1}@userii\endcsname
6311 }
```

\showglouseriii \showglouseriii{\label{}}

```
6312 \newcommand*{\showglouseriii}[1]{%
6313   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriii\endcsname
6314 }
```

\showglouseriv \showglouseriv{\label{}}

```
6315 \newcommand*{\showglouseriv}[1]{%
6316   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriv\endcsname
6317 }
```

\showglouserv \showglouserv{\label{}}

```
6318 \newcommand*{\showglouserv}[1]{%
6319   \expandafter\show\csname glo@\glsdetoklabel{#1}@userv\endcsname
6320 }
```

\showglouservi \showglouservi{\label{}}

```
6321 \newcommand*{\showglouservi}[1]{%
6322   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
6323 }
```

\showglouservi \showglouservi{\label{}}

```
6324 \newcommand*{\showglouservi}[1]{%
6325   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
6326 }
```

`\showglo`name `\showglo`name{*\label*}

```
6327 \newcommand*{\showglo
```

name}[1]{%
6328 \expandafter\show\csname glo@\glsdetoklabel{#1}@name\endcsname
6329 }

`\showglo`desc `\showglo`desc{*\label*}

```
6330 \newcommand*{\showglo
```

desc}[1]{%
6331 \expandafter\show\csname glo@\glsdetoklabel{#1}@desc\endcsname
6332 }

`\showglo`descplural `\showglo`descplural{*\label*}

```
6333 \newcommand*{\showglo
```

descplural}[1]{%
6334 \expandafter\show\csname glo@\glsdetoklabel{#1}@descplural\endcsname
6335 }

`\showglo`sort `\showglo`sort{*\label*}

```
6336 \newcommand*{\showglo
```

sort}[1]{%
6337 \expandafter\show\csname glo@\glsdetoklabel{#1}@sort\endcsname
6338 }

`\showglo`symbol `\showglo`symbol{*\label*}

```
6339 \newcommand*{\showglo
```

symbol}[1]{%
6340 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbol\endcsname
6341 }

`\showglo`symbolplural `\showglo`symbolplural{*\label*}

```
6342 \newcommand*{\showglo
```

symbolplural}[1]{%
6343 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolplural\endcsname
6344 }

`\showgloshort` `\showgloshort{<label>}`

```
6345 \newcommand*{\showgloshort}[1]{%
6346   \expandafter\show\csname glo@\glsdetoklabel{#1}@short\endcsname
6347 }
```

`\showglolong` `\showglolong{<label>}`

```
6348 \newcommand*{\showglolong}[1]{%
6349   \expandafter\show\csname glo@\glsdetoklabel{#1}@long\endcsname
6350 }
```

`\showgloindex` `\showgloindex{<label>}`

```
6351 \newcommand*{\showgloindex}[1]{%
6352   \expandafter\show\csname glo@\glsdetoklabel{#1}@index\endcsname
6353 }
```

`\showgloflag` `\showgloflag{<label>}`

```
6354 \newcommand*{\showgloflag}[1]{%
6355   \expandafter\show\csname ifglo@\glsdetoklabel{#1}@flag\endcsname
6356 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
6357 \newcommand*{\showacronymlists}{%
6358   \show\@glsacronymlists
6359 }
```

`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
6360 \newcommand*{\showglossaries}{%
6361   \show\@glo@types
6362 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
6363 \newcommand*{\showglossaryin}[1]{%
6364   \expandafter\show\csname @glotype@#1@in\endcsname
6365 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
6366 \newcommand*{\showglossaryout}[1]{%
6367   \expandafter\show\csname @glotype@#1@out\endcsname
6368 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
6369 \newcommand*{\showglossarytitle}[1]{%
6370   \expandafter\show\csname @glotype@#1@title\endcsname
6371 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
6372 \newcommand*{\showglossarycounter}[1]{%
6373   \expandafter\show\csname @glotype@#1@counter\endcsname
6374 }
```

`\showglossaryentries` `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
6375 \newcommand*{\showglossaryentries}[1]{%
6376   \expandafter\show\csname glolist@#1\endcsname
6377 }
```

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 Kindy 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.

```
6378 \csname ifglscompatible-2.07\endcsname
6379 \RequirePackage{glossaries-compatible-207}
6380 \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.

```
6381 \NeedsTeXFormat{LaTeX2e}
6382 \ProvidesPackage{glossaries-prefix}[2013/11/14 v4.0 (NLCT)]
```

Pass all options to glossaries:

```
6383 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
6384 \ProcessOptions
```

Load glossaries:

```
6385 \RequirePackage{glossaries}
```

Add the new keys:

```
6386 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
6387 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
6388 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
6389 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%
```

Add them to `\@gls@keymap`:

```
6390 \appto\@gls@keymap{,%
6391   {prefixfirst}{prefixfirst},%
6392   {prefixfirstplural}{prefixfirstplural},%
6393   {prefix}{prefix},%
6394   {prefixplural}{prefixplural}}%
6395 }
```

Set the default values:

```
6396 \appto\@newglossaryentryprehook{%
6397   \def\@glo@entryprefix{}%
6398   \def\@glo@entryprefixplural{}}
```



```

6399 \let\@glo@entryprefixfirst\@gls@default@value
6400 \let\@glo@entryprefixfirstplural\@gls@default@value
6401 }

```

Set the assignment code:

```

6402 \appto\@newglossaryentryposthook{%
6403   \gls@assign@field{}\@glo@label}{prefix}\@glo@entryprefix}%
6404   \gls@assign@field{}\@glo@label}{prefixplural}\@glo@entryprefixplural}%

```

If prefixfirst has not been supplied, make it the same as prefix.

```

6405   \expandafter\gls@assign@field\expandafter
6406     {\csname glo@\@glo@label @prefix\endcsname}\@glo@label}{prefixfirst}%
6407     {\@glo@entryprefixfirst}%

```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```

6408   \expandafter\gls@assign@field\expandafter
6409     {\csname glo@\@glo@label @prefixplural\endcsname}\@glo@label}%
6410     {prefixfirstplural}\@glo@entryprefixfirstplural}%
6411 }

```

Define commands to access these fields:

glsentryprefixfirst

```

6412 \newcommand*\{glsentryprefixfirst}[1]{\csuse{glo@#1@prefixfirst}}

```

ryprefixfirstplural

```

6413 \newcommand*\{glsentryprefixfirstplural}[1]{\csuse{glo@#1@prefixfirstplural}}

```

\glsentryprefix

```

6414 \newcommand*\{glsentryprefix}[1]{\csuse{glo@#1@prefix}}

```

lsentryprefixplural

```

6415 \newcommand*\{glsentryprefixplural}[1]{\csuse{glo@#1@prefixplural}}

```

Now for the initial upper case variants:

Glsentryprefixfirst

```

6416 \newrobustcmd*\{Glsentryprefixfirst}[1]{%
6417   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%
6418   \xmakefirstuc\@glo@text
6419 }

```

ryprefixfirstplural

```

6420 \newrobustcmd*\{Glsentryprefixfirstplural}[1]{%
6421   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
6422   \xmakefirstuc\@glo@text
6423 }

```

\Glsentryprefix

```
6424 \newrobustcmd*{\Glsentryprefix}[1]{%
6425   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
6426   \xmakefirstuc\@glo@text
6427 }
```

lsentryprefixplural

```
6428 \newrobustcmd*{\Glsentryprefixplural}[1]{%
6429   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
6430   \xmakefirstuc\@glo@text
6431 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
6432 \newcommand*{\ifglshasprefix}[3]{%
6433   \ifcempty{glo@#1@prefix}%
6434   {#3}%
6435   {#2}%
6436 }
```

ifglshasprefixplural

```
6437 \newcommand*{\ifglshasprefixplural}[3]{%
6438   \ifcempty{glo@#1@prefixplural}%
6439   {#3}%
6440   {#2}%
6441 }
```

ifglshasprefixfirst

```
6442 \newcommand*{\ifglshasprefixfirst}[3]{%
6443   \ifcempty{glo@#1@prefixfirst}%
6444   {#3}%
6445   {#2}%
6446 }
```

asprefixfirstplural

```
6447 \newcommand*{\ifglshasprefixfirstplural}[3]{%
6448   \ifcempty{glo@#1@prefixfirstplural}%
6449   {#3}%
6450   {#2}%
6451 }
```

Define commands that insert the prefix before commands like \gls:

\pgls

```
6452 \newrobustcmd{\pgls}{\@ifstar\@spgls\@pgls}
```

\@spgls Starred version.

```
6453 \newcommand*{\@spgls}[2][\@pgls@{hyper=false,#1}{#2}]
```

\@pgls Unstarred version.

```
6454 \newcommand*\@pgls}[2][]{%
6455   \new@ifnextchar[%
6456     {\@pgls@{#1}{#2}}%
6457     {\@pgls@{#1}{#2}[]}%
6458 }
```

\@pgls@ Read in the final optional argument:

```
6459 \def\@pgls@#1#2[#3]{%
6460   \glsdoifexists{#2}%
6461   {%
6462     \ifglsused{#2}%
6463     {%
6464       \glsentryprefix{#2}%
6465     }%
6466     {%
6467       \glsentryprefixfirst{#2}%
6468     }%
6469     \@gls@{#1}{#2}[#3]%
6470   }%
6471 }
```

Similarly for the plural version:

\pglsp1

```
6472 \newrobustcmd{\pglsp1}{\@ifstar\@spglsp1\@pglsp1}
```

\@spglsp1 Starred version.

```
6473 \newcommand*\@spglsp1}[2][]{\@pglsp1{hyper=false,#1}{#2}}
```

\@pglsp1 Unstarred version.

```
6474 \newcommand*\@pglsp1}[2][]{%
6475   \new@ifnextchar[%
6476     {\@pglsp1@{#1}{#2}}%
6477     {\@pglsp1@{#1}{#2}[]}%
6478 }
```

\@pglsp1@ Read in the final optional argument:

```
6479 \def\@pglsp1@#1#2[#3]{%
6480   \glsdoifexists{#2}%
6481   {%
6482     \ifglsused{#2}%
6483     {%
6484       \glsentryprefixplural{#2}%
6485     }%
6486     {%
6487       \glsentryprefixfirstplural{#2}%
6488     }%
6489     \@glspl@{#1}{#2}[#3]%
6490   }%
6491 }
```

```

6490 }%
6491 }

```

Now for the first letter upper case versions:

\Pgl's

```

6492 \newrobustcmd{\Pgl's}{\@ifstar\@sPgl's\@Pgl's}

```

\@sPgl's Starred version.

```

6493 \newcommand*{\@sPgl's}[2][\@Pgl's@{hyper=false,#1}{#2}]

```

\@Pgl's Unstarred version.

```

6494 \newcommand*{\@Pgl's}[2][\%
6495   \new@ifnextchar[%
6496   {\@Pgl's@{#1}{#2}}%
6497   {\@Pgl's@{#1}{#2}[]}%
6498 ]

```

\@Pgl's@ Read in the final optional argument:

```

6499 \def\@Pgl's@#1#2[#3]{%
6500   \gl'sdoifexists{#2}%
6501   {%
6502     \ifgl'sused{#2}%
6503     {%
6504       \ifgl'shasprefix{#2}%
6505       {%
6506         \Gl'sentryprefix{#2}%
6507         \@gl's@{#1}{#2}[#3]%
6508       }%
6509       {\@Gl's@{#1}{#2}[#3]}%
6510     }%
6511     {%
6512       \ifgl'shasprefixfirst{#2}%
6513       {%
6514         \Gl'sentryprefixfirst{#2}%
6515         \@gl's@{#1}{#2}[#3]%
6516       }%
6517       {\@Gl's@{#1}{#2}[#3]}%
6518     }%
6519   }%
6520 }

```

Similarly for the plural version:

\Pgl'spl

```

6521 \newrobustcmd{\Pgl'spl}{\@ifstar\@sPgl'spl\@Pgl'spl}

```

\@sPgl'spl Starred version.

```

6522 \newcommand*{\@sPgl'spl}[2][\@Pgl'spl@{hyper=false,#1}{#2}]

```

\@Pglsp1 Unstarred version.

```
6523 \newcommand*{\@Pglsp1}[2][]{%
6524   \new@ifnextchar[%
6525     {\@Pglsp1@{#1}{#2}}}%
6526     {\@Pglsp1@{#1}{#2}[]}%
6527 }
```

\@Pglsp1@ Read in the final optional argument:

```
6528 \def\@Pglsp1@#1#2[#3]{%
6529   \glsdoifexists{#2}%
6530   {%
6531     \ifglshasprefixplural{#2}%
6532     {%
6533       \ifglshasprefixplural{#2}%
6534       {%
6535         \Glsentryprefixplural{#2}%
6536         \@glsp1@{#1}{#2}[#3]%
6537       }%
6538       {\@Glspl@{#1}{#2}[#3]}%
6539     }%
6540     {%
6541       \ifglshasprefixfirstplural{#2}%
6542       {%
6543         \Glsentryprefixfirstplural{#2}%
6544         \@glsp1@{#1}{#2}[#3]%
6545       }%
6546       {\@Glspl@{#1}{#2}[#3]}%
6547     }%
6548   }%
6549 }
```

Finally the all upper case versions:

\PGLS

```
6550 \newrobustcmd{\PGLS}{\@ifstar\@sPGLS\@PGLS}
```

\@sPGLS Starred version.

```
6551 \newcommand*{\@sPGLS}[2][]{\@PGLS@{hyper=false,#1}{#2}}
```

\@PGLS Unstarred version.

```
6552 \newcommand*{\@PGLS}[2][]{%
6553   \new@ifnextchar[%
6554     {\@PGLS@{#1}{#2}}}%
6555     {\@PGLS@{#1}{#2}[]}%
6556 }
```

\@PGLS@ Read in the final optional argument:

```
6557 \def\@PGLS@#1#2[#3]{%
6558   \glsdoifexists{#2}%
6559 }
```

```

6559  {%
6560    \ifglused{#2}%
6561    {%
6562      \mfirstucMakeUppercase{\glentryprefix{#2}}%
6563    }%
6564    {%
6565      \mfirstucMakeUppercase{\glentryprefixfirst{#2}}%
6566    }%
6567    \@GLS@{#1}{#2}[#3]%
6568  }%
6569 }

```

Plural version:

\PGLSp1

```

6570 \newrobustcmd{\PGLSp1}{\@ifstar\@sPGLSp1\@PGLSp1}

```

\@sPGLSp1 Starred version.

```

6571 \newcommand*{\@sPGLSp1}[2][\@PGLSp1@{hyper=false,#1}{#2}]

```

\@PGLSp1 Unstarred version.

```

6572 \newcommand*{\@PGLSp1}[2][{%
6573   \new@ifnextchar[%
6574     {\@PGLSp1@{#1}{#2}}%
6575     {\@PGLSp1@{#1}{#2}[]}%
6576 }

```

\@PGLSp1@ Read in the final optional argument:

```

6577 \def\@PGLSp1@#1#2[#3]{%
6578   \glsoifexists{#2}%
6579   {%
6580     \ifglused{#2}%
6581     {%
6582       \mfirstucMakeUppercase{\glentryprefixplural{#2}}%
6583     }%
6584     {%
6585       \mfirstucMakeUppercase{\glentryprefixfirstplural{#2}}%
6586     }%
6587     \@GLSp1@{#1}{#2}[#3]%
6588   }%
6589 }

```

3 Mfirstuc Documented Code

```

6590 \NeedsTeXFormat{LaTeX2e}
6591 \ProvidesPackage{mfirstuc}[2013/11/04 v1.08 (NLCT)]

```

Requires etoolbox:

6592 \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`.

6593 \newif\if@glscs

6594 \newtoks\@glsmfirst

6595 \newtoks\@glsmrest

6596 \newrobustcmd*{\makefirstuc}[1]{%

6597 \def\gls@argi{#1}%

6598 \ifx\gls@argi\@empty

If the argument is empty, do nothing.

6599 \else

6600 \def\@gls@tmp{\ #1}%

6601 \@onelevel@sanitize\@gls@tmp

6602 \expandafter\@gls@checkcs\@gls@tmp\relax\relax

6603 \if@glscs

6604 \@gls@getbody #1{}\@nil

6605 \ifx\@gls@rest\@empty

6606 \glsmakefirstuc{#1}%

6607 \else

6608 \expandafter\@gls@split\@gls@rest\@nil

6609 \ifx\@gls@first\@empty

6610 \glsmakefirstuc{#1}%

6611 \else

6612 \expandafter\@glsmfirst\expandafter{\@gls@first}%

6613 \expandafter\@glsmrest\expandafter{\@gls@rest}%

6614 \edef\@gls@domfirstuc{\noexpand\@gls@body

6615 {\noexpand\glsmakefirstuc\the\@glsmfirst}%

6616 \the\@glsmrest}%

6617 \@gls@domfirstuc

6618 \fi

6619 \fi

6620 \else

6621 \glsmakefirstuc{#1}%

6622 \fi

6623 \fi

6624 }

Put first argument in `\@gls@first` and second argument in `\@gls@rest`:

6625 \def\@gls@split#1#2\@nil{%

```

6626 \def\@gls@first{#1}\def\@gls@rest{#2}%
6627 }

6628 \def\@gls@checkcs#1 #2#3\relax{%
6629 \def\@gls@argi{#1}\def\@gls@argii{#2}%
6630 \ifx\@gls@argi\@gls@argii
6631 \@gls@true
6632 \else
6633 \@gls@false
6634 \fi
6635 }

```

`\@gls@makefirstuc` Make first thing upper case:

```

6636 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}

```

`\mfirstucMakeUppercase` Allow user to replace `\MakeUppercase` with another case changing command.

```

6637 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}

```

`\glsmakefirstuc` Provide a user command to make it easier to customise.

```

6638 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in `\@gls@body`.

```

6639 \def\@gls@getbody#1#{\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoup up everything to `\@nil` and store in `\@gls@rest`:

```

6640 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

`\xmakefirstuc` Expand argument once before applying `\makefirstuc` (added v1.01).

```

6641 \newcommand*{\xmakefirstuc}[1]{%
6642 \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).

```

6643 \newrobustcmd*{\capitalisewords}[1]{%
6644 \def\gls@add@space{}%
6645 \mfu@capitalisewords#1 \@nil\mfu@endcap
6646 }

6647 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
6648 \def\mfu@cap@first{#1}%
6649 \def\mfu@cap@second{#2}%
6650 \gls@add@space
6651 \makefirstuc{#1}%
6652 \def\gls@add@space{ }%
6653 \ifx\mfu@cap@second\@nnil
6654 \let\next\mfu@cap\mfu@noop
6655 \else
6656 \let\next\mfu@cap\mfu@capitalisewords
6657 \fi

```



```

6658 \next@mfu@cap#2\mfu@endcap
6659 }
6660 \def\mfu@noop#1\mfu@endcap{}

```

`\xcapitalisewords` Short-cut command:

```

6661 \newcommand*\xcapitalisewords}[1]{%
6662 \expandafter\capitalisewords\expandafter{#1}%
6663 }

```

4 Glossary Styles

4.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```

6664 \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 hypertarget 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`

```

6665 \newcommand*\glsnavhyperlink}[3][\@glo@type]{%
6666 \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
6667 \@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`

```

6668 \newcommand*\glsnavhypertarget}[3][\@glo@type]{%

```

Add this group to the aux file for re-run check.

```

6669 \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%

```

Add the target.

```

6670 \@gls@target{glsn:#1@#2}{#3}%

```

Check list of known groups to determine if a re-run is required.

```

6671 \expandafter\let
6672 \expandafter\@gls@list\csname @gls@hypergroup@#1\endcsname

```

Iterate through list and terminate loop if this group is found.

```
6673 \@for\@gls@elem:=\@gls@list\do{%
6674 \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
```

Check if list terminated prematurely.

```
6675 \if@endfor
6676 \else
```

This group was not included in the list, so issue a warning.

```
6677 \GlossariesWarningNoLine{Navigation panel
6678 for glossary type ‘#1’^^Jmissing group ‘#2’}%
6679 \gdef\gls@hypergroup@rerun{%
6680 \GlossariesWarningNoLine{Navigation panel
6681 has changed. Rerun LaTeX}}%
6682 \fi
6683 }
```

`\gls@hypergroup@rerun` Give a warning at the end if re-run required

```
6684 \let\gls@hypergroup@rerun\relax
6685 \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.

```
6686 \newcommand*{\@gls@hypergroup}[2]{%
6687 \@ifundefined{\@gls@hypergroup@list@#1}{%
6688 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{#2}%
6689 }{%
6690 \expandafter\let\expandafter\@gls@tmp
6691 \csname @gls@hypergroup@list@#1\endcsname
6692 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{%
6693 \@gls@tmp,#2}%
6694 }%
6695 }
```

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`

```
6696 \newcommand*{\glsnavigation}{%
6697 \def\@gls@between{}%
6698 \@ifundefined{\@gls@hypergroup@list@\@gls@type}{%
6699 \def\@gls@list{}%
6700 }{%
```

```

6701 \expandafter\let\expandafter\@gls@list
6702 \csname @gls@hypergrouplist@\@glo@type\endcsname
6703 }%
6704 \@for\@gls@tmp:=\@gls@list\do{%
6705 \@gls@between

6706 \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%
6707 \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
6708 \let\@gls@between\glshypernavsep%
6709 }%
6710 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```
6711 \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`

```

6712 \newcommand*{\glssymbolnav}{%
6713 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
6714 \glshypernavsep
6715 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
6716 \glshypernavsep
6717 }

```

4.2 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.

```
6718 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]
```

`inline` Define the inline style.

```
6719 \newglossarystyle{inline}{%
```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```

6720 \renewenvironment{theglossary}%
6721 {%
6722 \def\gls@inlinesep{}%
6723 \def\gls@inlinesubsep{}%
6724 \def\gls@inlinepostchild{}%
6725 }%
6726 {\glspostinline}%

```

No header:

```
6727 \renewcommand*{\glossaryheader}{}%
```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```
6728 \renewcommand*{\glsgroupheading}[1]{}%
```

Just display separator followed by name and description:

```
6729 \renewcommand{\glossentry}[2]{%
6730   \glsinlinedopostchild
6731   \gls@inlinesep
6732   \glsentryitem{##1}%
6733   \glsinlinenameformat{##1}{%
6734     \glossentryname{##1}%
6735   }%
6736   \ifglsdescsuppressed{##1}%
6737   {%
6738     \glsinlineemptydescformat
6739     {%
6740       \glossentrysymbol{##1}%
6741     }%
6742     {%
6743       ##2%
6744     }%
6745   }%
6746   {%
6747     \ifglsdesc{##1}%
6748     {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
6749     {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
6750   }%
6751   \ifglschildren{##1}%
6752   {%
6753     \glsresetsubentrycounter
6754     \glsinlineparentchildseparator
6755     \def\gls@inlinesubsep{}%
6756     \def\gls@inlinepostchild{\glsinlinepostchild}%
6757   }%
6758   {%}%
6759   \def\gls@inlinesep{\glsinlineseparator}%
6760 }%
```

Sub-entries display description:

```
6761 \renewcommand{\subglossentry}[3]{%
6762   \gls@inlinesubsep%
6763   \glsinlinesubnameformat{##2}{%
6764     \glossentryname{##2}%
6765   }%
6766   \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
6767   \def\gls@inlinesubsep{\glsinlinesubseparator}%
6768 }%
```

Nothing special between groups:

```
6769 \renewcommand*{\glsgroupskip}{}%
```

```

6770 }

\glsinlinedopostchild
6771 \newcommand*{\glsinlinedopostchild}{%
6772     \gls@inlinepostchild
6773     \def\gls@inlinepostchild{}}%
6774 }

\glsinlineseparator  Separator to use between entries.
6775 \newcommand*{\glsinlineseparator}{;\space}

\glsinlinesubseparator  Separator to use between sub-entries.
6776 \newcommand*{\glsinlinesubseparator}{,\space}

\glsinlinparentchildseparator  Separator to use between parent and children.
6777 \newcommand*{\glsinlinparentchildseparator}{:\space}

\glsinlinepostchild  Hook to use between child and next entry
6778 \newcommand*{\glsinlinepostchild}{}

\glspostinline  Terminator for inline glossary.
6779 \newcommand*{\glspostinline}{\glspostdescription\space}

\glsinlinenameformat  Formats the name of the entry (first argument label, second argument name):
6780 \newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}

\glsinlinedescformat  Formats the entry's description, symbol and location list:
6781 \newcommand*{\glsinlinedescformat}[3]{\space#1}

\glsinlineemptydescformat  Formats the entry's symbol and location list when the description is empty:
6782 \newcommand*{\glsinlineemptydescformat}[2]{}

\glsinlinesubnameformat  Formats the name of the subentry (first argument label, second argument
name):
6783 \newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}

\glsinlinesubdescformat  Formats the subentry's description, symbol and location list:
6784 \newcommand*{\glsinlinesubdescformat}[3]{#1}

```

4.3 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.

```
6785 \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.

```
6786 \newglossarystyle{list}{%
```

Use description environment:

```
6787 \renewenvironment{theglossary}{%
```

```
6788 {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
6789 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6790 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
6791 \renewcommand*{\glossentry}[2]{%
```

```
6792 \item[\glsentryitem{##1}]%
```

```
6793 \glstarget{##1}{\glossentryname{##1}}]
```

```
6794 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
6795 \renewcommand*{\subglossentry}[3]{%
```

```
6796 \glssubentryitem{##2}%
```

```
6797 \glstarget{##2}{\strut}%
```

```
6798 \glossentrydesc{##2}\glspostdescription\space ##3.}%
```

```
6799 % \end{macrocode}
```

```
6800 % Add vertical space between groups:
```

```
6801 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
```

```
6802 % \begin{macrocode}
```

```
6803 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
```

```
6804 }
```

`listgroup` The listgroup style is like the list style, but the glossary groups have headings.

```
6805 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
6806 \setglossarystyle{list}%
```

Each group has a heading:

```
6807 \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.

```
6808 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
6809 \setglossarystyle{list}%
```

Add navigation links at the start of the environment:

```
6810 \renewcommand*{\glossaryheader}{%  
6811 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
6812 \renewcommand*{\glsgroupheading}[1]{%  
6813 \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.

```
6814 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
6815 \setglossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
6816 \renewcommand*{\glossentry}[2]{%  
6817 \item[\glsentryitem{##1}%  
6818 \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.

```
6819 \mbox{}\par\nobreak\@afterheading  
6820 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
6821 \renewcommand{\subglossentry}[3]{%  
6822 \par  
6823 \glssubentryitem{##2}%  
6824 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%  
6825 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
6826 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
6827 \setglossarystyle{altlist}%
```

Each group has a heading:

```
6828 \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.

```
6829 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
6830 \setglossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
6831 \renewcommand*{\glossaryheader}{%
6832 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
6833 \renewcommand*{\glsgroupheading}[1]{%
6834 \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.

```
6835 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
6836 \setglossarystyle{list}%
```

Each main (level 0) entry starts a new item:

```
6837 \renewcommand*{\glossentry}[2]{%
6838 \item[]\makebox[\glslistdottedwidth][l]{%
6839 \glssentryitem{##1}%
6840 \glstarget{##1}{\glossentryname{##1}}%
6841 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
6842 \renewcommand*{\subglossentry}[3]{%
6843 \item[]\makebox[\glslistdottedwidth][l]{%
6844 \glssubentryitem{##2}%
6845 \glstarget{##2}{\glossentryname{##2}}%
6846 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
6847 }
```

`\glslistdottedwidth`

```
6848 \newlength\glslistdottedwidth
6849 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
6850 \newglossarystyle{sublistdotted}{%
```

Base it on the listdotted style:

```
6851 \setglossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
6852 \renewcommand*{\glossentry}[2]{%
6853 \item[\glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}}]}%
6854 }
```


4.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
6855 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
6856 \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`.)

```
6857 \@ifundefined{glsdescwidth}{%
6858   \newlength{glsdescwidth
6859   \setlength{glsdescwidth}{0.6\hsize}
6860 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
6861 \@ifundefined{glspagelistwidth}{%
6862   \newlength{glspagelistwidth
6863   \setlength{glspagelistwidth}{0.1\hsize}
6864 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
6865 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
6866   \renewenvironment{theglossary}%
6867   {\begin{longtable}{lp{glsdescwidth}}}%
6868   {\end{longtable}}%
```

Do nothing at the start of the environment:

```
6869   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
6870   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
6871   \renewcommand{\glossentry}[2]{%
6872     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6873     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
6874   }%
```

Sub entries displayed on the following row without the name:

```
6875   \renewcommand{\subglossentry}[3]{%
6876     &
6877     \glssubentryitem{##2}%
6878     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
6879     ##3\tabularnewline
6880   }%
```

Blank row between groups:

```
6881 \renewcommand*{\glsnoglobskip}{\ifglsnogroupskip\else &
6882 \tabularnewline\fi}%
6883 }
```

longborder The longborder style is like the above, but with horizontal and vertical lines:

```
6884 \newglossarystyle{longborder}{%
```

Base it on the glostylelong style:

```
6885 \setglossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
6886 \renewenvironment{theglossary}{%
6887 \begin{longtable}{|l|p{\glsdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
6888 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6889 }
```

longheader The longheader style is like the long style but with a header:

```
6890 \newglossarystyle{longheader}{%
```

Base it on the glostylelong style:

```
6891 \setglossarystyle{long}%
```

Set the table's header:

```
6892 \renewcommand*{\glossaryheader}{%
6893 \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%
6894 }
```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```
6895 \newglossarystyle{longheaderborder}{%
```

Base it on the glostylelongborder style:

```
6896 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
6897 \renewcommand*{\glossaryheader}{%
6898 \hline\bfseries \entryname & \bfseries
6899 \descriptionname\tabularnewline\hline
6900 \endhead
6901 \hline\endfoot}%
6902 }
```

long3col The long3col style is like long but with 3 columns

```
6903 \newglossarystyle{long3col}{%
```

Use a longtable with 3 columns:

```
6904 \renewenvironment{theglossary}{%
6905 {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
6906 {\end{longtable}}%
```

No table header:

```
6907 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
6908 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
6909 \renewcommand{\glossentry}[2]{%
6910   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6911   \glossentrydesc{##1} & ##2\tabularnewline
6912 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
6913 \renewcommand{\subglossentry}[3]{%
6914   &
6915   \glssubentryitem{##2}%
6916   \glstarget{##2}{\strut}\glossentrydesc{##2} &
6917   ##3\tabularnewline
6918 }%
```

Blank row between groups:

```
6919 \renewcommand*{\glsgroupskip}{%
6920   \ifglsnogroupskip\else & &\tabularnewline\fi}%
6921 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```
6922 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```
6923 \setglossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
6924 \renewenvironment{theglossary}%
6925   {\begin{longtable}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
6926   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
6927 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6928 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```
6929 \newglossarystyle{long3colheader}{%
```

Base it on the glostylelong3col style:

```
6930 \setglossarystyle{long3col}%
```

Set the table's header:

```
6931 \renewcommand*{\glossaryheader}{%
6932   \bfseries\entryname&\bfseries\descriptionname&
6933   \bfseries\pagelistname\tabularnewline\endhead}%
6934 }
```

long3colheaderborder The long3colheaderborder style is like the above but with a border

```
6935 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
6936 \setglossarystyle{long3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
6937 \renewcommand*{\glossaryheader}{%
6938 \hline
6939 \bfseries\entryname&\bfseries\descriptionname&
6940 \bfseries\pagelistname\tabularnewline\hline\endhead
6941 \hline\endfoot}%
6942 }
```

long4col The long4col style has four columns where the third column contains the value of the associated symbol key.

```
6943 \newglossarystyle{long4col}{%
```

Use a longtable with 4 columns:

```
6944 \renewenvironment{theglossary}%
6945 {\begin{longtable}{llll}}%
6946 {\end{longtable}}%
```

No table header:

```
6947 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6948 \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):

```
6949 \renewcommand{\glossentry}[2]{%
6950 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6951 \glossentrydesc{##1} &
6952 \glossentrysymbol{##1} &
6953 ##2\tabularnewline
6954 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
6955 \renewcommand{\subglossentry}[3]{%
6956 &
6957 \glssubentryitem{##2}%
6958 \glstarget{##2}{\strut}\glossentrydesc{##2} &
6959 \glossentrysymbol{##2} & ##3\tabularnewline
6960 }%
```

Blank row between groups:

```
6961 \renewcommand*{\glsgroupskip}{%
6962 \ifglsgnigroupskip\else & & \tabularnewline\fi}%
6963 }
```

`long4colheader` The `long4colheader` style is like `long4col` but with a header row.

```
6964 \newglossarystyle{long4colheader}{%
      Base it on the glostylelong4col style:
6965   \setglossarystyle{long4col}%
      Table has a header:
6966   \renewcommand*{\glossaryheader}{%
6967     \bfseries\entryname&\bfseries\descriptionname&
6968     \bfseries \symbolname&
6969     \bfseries\pagelistname\tabularnewline\endhead}%
6970 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
6971 \newglossarystyle{long4colborder}{%
      Base it on the glostylelong4col style:
6972   \setglossarystyle{long4col}%
      Use a longtable with 4 columns surrounded by vertical lines:
6973   \renewenvironment{theglossary}%
6974     {\begin{longtable}{|l|l|l|l|}}%
6975     {\end{longtable}}%
      Add horizontal lines to the head and foot of the table:
6976   \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6977 }
```

`long4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
6978 \newglossarystyle{long4colheaderborder}{%
      Base it on the glostylelong4col style:
6979   \setglossarystyle{long4col}%
      Use a longtable with 4 columns surrounded by vertical lines:
6980   \renewenvironment{theglossary}%
6981     {\begin{longtable}{|l|l|l|l|}}%
6982     {\end{longtable}}%
      Add table header and horizontal line at the table's foot:
6983   \renewcommand*{\glossaryheader}{%
6984     \hline\bfseries\entryname&\bfseries\descriptionname&
6985     \bfseries \symbolname&
6986     \bfseries\pagelistname\tabularnewline\hline\endhead
6987     \hline\endfoot}%
6988 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
6989 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
6990 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6991 \renewenvironment{theglossary}%  
6992 {\begin{longtable}{lp{\glstdescwidth}lp{\glspagelistwidth}}}%  
6993 {\end{longtable}}%  
6994 }
```

`altlong4colheader` The `altlong4colheader` style is like `altlong4col` but with a header row.

```
6995 \newglossarystyle{altlong4colheader}{%
```

Base it on the `glostylelong4colheader` style:

```
6996 \setglossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6997 \renewenvironment{theglossary}%  
6998 {\begin{longtable}{lp{\glstdescwidth}lp{\glspagelistwidth}}}%  
6999 {\end{longtable}}%  
7000 }
```

`altlong4colborder` The `altlong4colborder` style is like `altlong4col` but with a border.

```
7001 \newglossarystyle{altlong4colborder}{%
```

Base it on the `glostylelong4colborder` style:

```
7002 \setglossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7003 \renewenvironment{theglossary}%  
7004 {\begin{longtable}{|lp{\glstdescwidth}|lp{\glspagelistwidth}|}}%  
7005 {\end{longtable}}%  
7006 }
```

`ong4colheaderborder` The `altlong4colheaderborder` style is like the above but with a header as well as a border.

```
7007 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the `glostylelong4colheaderborder` style:

```
7008 \setglossarystyle{long4colheaderborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7009 \renewenvironment{theglossary}%  
7010 {\begin{longtable}{|lp{\glstdescwidth}|lp{\glspagelistwidth}|}}%  
7011 {\end{longtable}}%  
7012 }
```

4.5 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.

7013 \ProvidesPackage{glossary-longragged}[2013/11/14 v4.0 (NLCT)]

Requires the package:

7014 \RequirePackage{array}

Requires the package:

7015 \RequirePackage{longtable}

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
7016 \@ifundefined{glsdescwidth}{%
7017   \newlength{glsdescwidth
7018   \setlength{glsdescwidth}{0.6\hsize}
7019 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
7020 \@ifundefined{glspagelistwidth}{%
7021   \newlength{glspagelistwidth
7022   \setlength{glspagelistwidth}{0.1\hsize}
7023 }{}
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

7024 \newglossarystyle{longragged}{%

Use longtable with two columns:

```
7025   \renewenvironment{theglossary}{%
7026     {\begin{longtable}{l>{\raggedright}p{glsdescwidth}}}%
7027     {\end{longtable}}}%
```

Do nothing at the start of the environment:

7028 \renewcommand*{\glossaryheader}{}%

No heading between groups:

7029 \renewcommand*{\glsgroupheading}[1]{}%

Main (level 0) entries displayed in a row:

```
7030 \renewcommand{\glossentry}[2]{%
7031   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7032   \glossentrydesc{##1}\glspostdescription\space ##2%
7033   \tabularnewline
7034 }%
```

Sub entries displayed on the following row without the name:

```

7035 \renewcommand{\subglossentry}[3]{%
7036     &
7037     \glssubentryitem{##2}%
7038     \glstarget{##2}{\strut}\glossentrydesc{##2}%
7039     \glspostdescription\space ##3%
7040     \tabularnewline
7041 }%

```

Blank row between groups:

```

7042 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
7043 }

```

`longraggedborder` The `longraggedborder` style is like the above, but with horizontal and vertical lines:

7044 \newglossarystyle{longraggedborder}{%

Base it on the `glostylelongragged` style:

```
7045 \setglossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```

7046 \renewenvironment{theglossary}{%
7047 \begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|}%
7048 {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```
7049 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7050 }
```

`longraggedheader` The `longraggedheader` style is like the `longragged` style but with a header:

7051 \newglossarystyle{longraggedheader}{%

Base it on the `glostylelongragged` style:

```
7052 \setglossarystyle{longragged}%
```

Set the table's header:

```

7053 \renewcommand*{\glossaryheader}{%
7054     \bfseries \entryname & \bfseries \descriptionname
7055     \tabularnewline\endhead}%
7056 }

```

`longraggedheaderborder` The `longraggedheaderborder` style is like the `longragged` style but with a header and border:

7057 \newglossarystyle{longraggedheaderborder}{%

Base it on the `glostylelongraggedborder` style:

```
7058 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```

7059 \renewcommand*{\glossaryheader}{%
7060 \hline\bfseries \entryname & \bfseries \descriptionname

```



```

7061 \tabularnewline\hline
7062 \endhead
7063 \hline\endfoot}%
7064 }

```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```

7065 \newglossarystyle{longragged3col}{%

```

Use a longtable with 3 columns:

```

7066 \renewenvironment{theglossary}%
7067 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
7068 >{\raggedright}p{\glspagelistwidth}}}%
7069 {\end{longtable}}%

```

No table header:

```

7070 \renewcommand*{\glossaryheader}{}%

```

No headings between groups:

```

7071 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7072 \renewcommand{\glossentry}[2]{%
7073 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7074 \glossentrydesc{##1} & ##2\tabularnewline
7075 }%

```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```

7076 \renewcommand{\subglossentry}[3]{%
7077 &
7078 \glssubentryitem{##2}%
7079 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7080 ##3\tabularnewline
7081 }%

```

Blank row between groups:

```

7082 \renewcommand*{\glsgroupskip}{%
7083 \ifglsnogroupskip\else & &\tabularnewline\fi}%
7084 }

```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```

7085 \newglossarystyle{longragged3colborder}{%

```

Base it on the `glostylelongragged3col` style:

```

7086 \setglossarystyle{longragged3col}%

```

Use a longtable with 3 columns with vertical lines around them:

```

7087 \renewenvironment{theglossary}%
7088 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
7089 >{\raggedright}p{\glspagelistwidth}|}%
7090 {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```
7091 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7092 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7093 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7094 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
7095 \renewcommand*{\glossaryheader}{%
7096 \bfseries\entryname&\bfseries\descriptionname&
7097 \bfseries\pagelistname\tabularnewline\endhead}%
7098 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7099 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7100 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7101 \renewcommand*{\glossaryheader}{%
7102 \hline
7103 \bfseries\entryname&\bfseries\descriptionname&
7104 \bfseries\pagelistname\tabularnewline\hline\endhead
7105 \hline\endfoot}%
7106 }
```

`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.

```
7107 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7108 \renewenvironment{theglossary}%
7109 {\begin{longtable}{l>{\raggedright}p{\glstdescwidth}l%
7110 >{\raggedright}p{\glspagelistwidth}}}%
7111 {\end{longtable}}%
```

No table header:

```
7112 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7113 \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):

```
7114 \renewcommand{\glossentry}[2]{}%
```

```

7115 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7116 \glossentrydesc{##1} & \glossentrydesc{##1} &
7117 ##2\tabularnewline
7118 }%

```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```

7119 \renewcommand{\subglossentry}[3]{%
7120 &
7121 \glssubentryitem{##2}%
7122 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7123 \glossentrysymbol{##2} & ##3\tabularnewline
7124 }%

```

Blank row between groups:

```

7125 \renewcommand*\glsgroupskip{%
7126 \ifglsgroupskip\else & & \tabularnewline\fi}%
7127 }

```

`longragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```

7128 \newglossarystyle{altlongragged4colheader}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7129 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7130 \renewenvironment{theglossary}%
7131 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7132 >{\raggedright}p{\glspagelistwidth}}}%
7133 {\end{longtable}}%

```

Table has a header:

```

7134 \renewcommand*\glossaryheader{%
7135 \bfseries\entryname&\bfseries\descriptionname&
7136 \bfseries \symbolname&
7137 \bfseries\pagelistname\tabularnewline\endhead}%
7138 }

```

`longragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```

7139 \newglossarystyle{altlongragged4colborder}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7140 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7141 \renewenvironment{theglossary}%
7142 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
7143 >{\raggedright}p{\glspagelistwidth}|}%
7144 {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```
7145 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7146 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
7147 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7148 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7149 \renewenvironment{theglossary}%
7150 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%
7151 >{\raggedright}p{\glspagelistwidth}|}%
7152 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7153 \renewcommand*{\glossaryheader}{%
7154 \hline\bfseries\entryname&\bfseries\descriptionname&
7155 \bfseries \symbolname&
7156 \bfseries\pagelistname\tabularnewline\hline\endhead
7157 \hline\endfoot}%
7158 }
```

4.6 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.

```
7159 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
7160 \RequirePackage{multicol}
7161 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
7162 \newcommand*{\glsmcols}{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.)

```
7163 \newglossarystyle{mcolindex}{%
7164 \setglossarystyle{index}%
7165 \renewenvironment{theglossary}%
7166 {%
```

```

7167 \begin{multicols}{\glsmcols}
7168 \setlength{\parindent}{0pt}%
7169 \setlength{\parskip}{0pt plus 0.3pt}%
7170 \let\item\@idxitem}%
7171 {\end{multicols}}%
7172 }

```

mcolindexgroup As mcolindex but has headings:

```

7173 \newglossarystyle{mcolindexgroup}{%
7174 \setglossarystyle{mcolindex}%
7175 \renewcommand*{\glsgroupheading}[1]{%
7176 \item\textbf{\glsgroupheading{##1}}\indexspace}%
7177 }

```

mcolindexhypergroup The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

7178 \newglossarystyle{mcolindexhypergroup}{%
  Base it on the glostylemcolindex style:
7179 \setglossarystyle{mcolindex}%
  Put navigation links to the groups at the start of the glossary:
7180 \renewcommand*{\glossaryheader}{%
7181 \item\textbf{\glshypernavigation}\indexspace}%
  Add a heading for each group (with a target). The group's title is in bold followed
  by a vertical gap.
7182 \renewcommand*{\glsgroupheading}[1]{%
7183 \item\textbf{\glshypernavhypertarget{##1}}{\glsgroupheading{##1}}}%
7184 \indexspace}%
7185 }

```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

7186 \newglossarystyle{mcoltree}{%
7187 \setglossarystyle{tree}%
7188 \renewenvironment{theglossary}%
7189 {%
7190 \begin{multicols}{\glsmcols}
7191 \setlength{\parindent}{0pt}%
7192 \setlength{\parskip}{0pt plus 0.3pt}%
7193 }%
7194 {\end{multicols}}%
7195 }

```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```

7196 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
7197 \setglossarystyle{mcoltree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```
7198 \renewcommand{\glsgroupheading}[1]{\par
7199 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7200 }
```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
7201 \newglossarystyle{mcoltreehypergroup}{%
```

Base it on the glostylemcoltree style:

```
7202 \setglossarystyle{mcoltree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
7203 \renewcommand*{\glossaryheader}{%
7204 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7205 \renewcommand*{\glsgroupheading}[1]{%
7206 \par\noindent
7207 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7208 \indexspace}%
7209 }
```

mcoltreenoname Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```
7210 \newglossarystyle{mcoltreenoname}{%
7211 \setglossarystyle{treenoname}%
7212 \renewenvironment{theglossary}%
7213 {%
7214 \begin{multicols}{\glsmcols}
7215 \setlength{\parindent}{0pt}%
7216 \setlength{\parskip}{0pt plus 0.3pt}%
7217 }%
7218 {\end{multicols}}}%
7219 }
```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```
7220 \newglossarystyle{mcoltreenonamegroup}{%
```

Base it on the glostylemcoltreenoname style:

```
7221 \setglossarystyle{mcoltreenoname}%
```

Give each group a heading:

```
7222 \renewcommand{\glsgroupheading}[1]{\par
7223 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7224 }
```

treenonamehypergroup The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```
7225 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreename` style:

```
7226 \setglossarystyle{mcoltreename}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7227 \renewcommand*{\glossaryheader}{%
```

```
7228 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7229 \renewcommand*{\glsgroupheading}[1]{%
```

```
7230 \par\noindent
```

```
7231 \textbf{\glsnavigationhypertarget{##1}{\glsgroupheading{##1}}}\par
```

```
7232 \indexspace}%
```

```
7233 }
```

`mcolalttree` Multi-column index style. Same as the `alttree`, but puts the glossary in multiple columns.

```
7234 \newglossarystyle{mcolalttree}{%
```

```
7235 \setglossarystyle{alttree}%
```

```
7236 \renewenvironment{theglossary}%
```

```
7237 {%
```

```
7238 \begin{multicols}{\glsmcols}
```

```
7239 \def\@gls@prevlevel{-1}%
```

```
7240 \mbox{}\par
```

```
7241 }%
```

```
7242 {\par\end{multicols}}%
```

```
7243 }
```

`mcolalttreegroup` Like the `mcolalttree` style but the glossary groups have headings.

```
7244 \newglossarystyle{mcolalttreegroup}{%
```

Base it on the `glostylemcolalttree` style:

```
7245 \setglossarystyle{mcolalttree}%
```

Give each group a heading.

```
7246 \renewcommand{\glsgroupheading}[1]{\par
```

```
7247 \def\@gls@prevlevel{-1}%
```

```
7248 \hangindent0pt\relax
```

```
7249 \parindent0pt\relax
```

```
7250 \textbf{\glsgroupheading{##1}}\par\indexspace}%
```

```
7251 }
```

`olalttreehypergroup` The `mcolalttreehypergroup` style is like the `mcolalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7252 \newglossarystyle{mcolalttreehypergroup}{%
```

Base it on the `glostylemcolalttree` style:

```
7253 \setglossarystyle{mcolalttree}%
```

Put the navigation links in the header

```
7254 \renewcommand*{\glossaryheader}{%  
7255 \par  
7256 \def\@gls@prevlevel{-1}%  
7257 \hangindent0pt\relax  
7258 \parindent0pt\relax  
7259 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7260 \renewcommand*{\glsgroupheading}[1]{%  
7261 \par  
7262 \def\@gls@prevlevel{-1}%  
7263 \hangindent0pt\relax  
7264 \parindent0pt\relax  
7265 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par  
7266 \indexspace}}
```

4.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
7267 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7268 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined if has been loaded.

```
7269 \@ifundefined{glsdescwidth}{%  
7270 \newlength\glsdescwidth  
7271 \setlength{\glsdescwidth}{0.6\hsize}  
7272 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
7273 \@ifundefined{glspagelistwidth}{%  
7274 \newlength\glspagelistwidth  
7275 \setlength{\glspagelistwidth}{0.1\hsize}  
7276 }{}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
7277 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7278 \renewenvironment{theglossary}{%  
7279 {\tablehead{}\tabletail{}}%  
7280 \begin{supertabular}{lp{\glsdescwidth}}}%  
7281 {\end{supertabular}}%
```


Do nothing at the start of the table:

```
7282 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7283 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7284 \renewcommand{\glossentry}[2]{%
7285   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7286   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7287 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
7288 \renewcommand{\subglossentry}[3]{%
7289   &
7290   \glssubentryitem{##2}%
7291   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7292   ##3\tabularnewline
7293 }%
```

Blank row between groups:

```
7294 \renewcommand*{\glsgroupskip}{}%
7295 \ifglsgnognogroupskip\else & \tabularnewline\fi}%
7296 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
7297 \newglossarystyle{superborder}{}%
```

Base it on the `glostylesuper` style:

```
7298 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
7299 \renewenvironment{theglossary}%
7300   {\tablehead{\hline}\tabletail{\hline}%
7301   \begin{supertabular}{|l|p{\glsdescwidth}|}%
7302   {\end{supertabular}}%
7303 }
```

superheader The superheader style is like the super style, but with a header:

```
7304 \newglossarystyle{superheader}{}%
```

Base it on the `glostylesuper` style:

```
7305 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
7306 \renewenvironment{theglossary}%
7307   {\tablehead{\bfseries \entryname &
7308   \bfseries\descriptionname\tabularnewline}%

```

```

7309 \tabletail{ }%
7310 \begin{supertabular}{lp{\glsgdescwidth}}%
7311 {\end{supertabular}}%
7312 }

```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
7313 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
7314 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

7315 \renewenvironment{theglossary}%
7316 {\tablehead{\hline\bfseries \entryname &
7317 \bfseries \descriptionname\tabularnewline\hline}%
7318 \tabletail{\hline}
7319 \begin{supertabular}{|lp{\glsgdescwidth}|}%
7320 {\end{supertabular}}%
7321 }

```

super3col The super3col style is like the super style, but with 3 columns:

```
7322 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

7323 \renewenvironment{theglossary}%
7324 {\tablehead{} \tabletail{ }%
7325 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
7326 {\end{supertabular}}%

```

Do nothing at the start of the table:

```
7327 \renewcommand*{\glossaryheader}{ }%
```

No group headings:

```
7328 \renewcommand*{\glsgroupheading}[1]{ }%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7329 \renewcommand{\glossentry}[2]{%
7330 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7331 \glossentrydesc{##1} & ##2\tabularnewline
7332 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

7333 \renewcommand{\subglossentry}[3]{%
7334 &
7335 \glssubentryitem{##2}%
7336 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7337 ##3\tabularnewline
7338 }%

```

Blank row between groups:

```
7339 \renewcommand*{\glsgroupskip}{%  
7340 \ifglsnogroupskip\else & &\tabularnewline\fi}%  
7341 }
```

super3colborder The **super3colborder** style is like the **super3col** style, but with a border:

```
7342 \newglossarystyle{super3colborder}{%  
Base it on the glostylesuper3col style:  
7343 \setglossarystyle{super3col}%  
Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:  
7344 \renewenvironment{theglossary}%  
7345 {\tablehead{\hline}\tabletail{\hline}%  
7346 \begin{supertabular}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%  
7347 {\end{supertabular}}%  
7348 }
```

super3colheader The **super3colheader** style is like the **super3col** style but with a header row:

```
7349 \newglossarystyle{super3colheader}{%  
Base it on the glostylesuper3col style:  
7350 \setglossarystyle{super3col}%  
Put the glossary in a supertabular environment with three columns, a header and no tail:  
7351 \renewenvironment{theglossary}%  
7352 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
7353 \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
7354 \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}%  
7355 {\end{supertabular}}%  
7356 }
```

super3colheaderborder The **super3colheaderborder** style is like the **super3col** style but with a header and border:

```
7357 \newglossarystyle{super3colheaderborder}{%  
Base it on the glostylesuper3colborder style:  
7358 \setglossarystyle{super3colborder}%  
Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:  
7359 \renewenvironment{theglossary}%  
7360 {\tablehead{\hline  
7361 \bfseries\entryname&\bfseries\descriptionname&  
7362 \bfseries\pagelistname\tabularnewline\hline}%  
7363 \tabletail{\hline}%  
7364 \begin{supertabular}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%  
7365 {\end{supertabular}}%  
7366 }
```

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.

```
7367 \newglossarystyle{super4col}{%
```

Put the glossary in a **supertabular** environment with four columns and no head or tail:

```
7368 \renewenvironment{theglossary}{%
7369 {\tablehead{}\tabletail}%
7370 \begin{supertabular}{llll}}{%
7371 \end{supertabular}}%
```

Do nothing at the start of the table:

```
7372 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7373 \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:

```
7374 \renewcommand{\glossentry}[2]{%
7375 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7376 \glossentrydesc{##1} &
7377 \glossentrysymbol{##1} & ##3\tabularnewline
7378 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
7379 \renewcommand{\subglossentry}[3]{%
7380 &
7381 \glssubentryitem{##2}%
7382 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7383 \glossentrysymbol{##2} & ##3\tabularnewline
7384 }%
```

Blank row between groups:

```
7385 \renewcommand*{\glsgroupskip}{%
7386 \ifglsnogroupskip\else & & \tabularnewline\fi}%
7387 }
```

super4colheader The **super4colheader** style is like the **super4col** but with a header row.

```
7388 \newglossarystyle{super4colheader}{%
```

Base it on the **glostylessuper4col** style:

```
7389 \setglossarystyle{super4col}%
```

Put the glossary in a **supertabular** environment with four columns, a header and no tail:

```
7390 \renewenvironment{theglossary}{%
7391 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7392 \bfseries\symbolname &
7393 \bfseries\pagelistname\tablearnewline}}%
```

```

7394     \tabletail{}}%
7395     \begin{supertabular}{|l|l|l|l|}%
7396     {\end{supertabular}}}%
7397 }

```

super4colborder The **super4colborder** style is like the **super4col** but with a border.

```

7398 \newglossarystyle{super4colborder}{%

```

Base it on the **glostylesuper4col** style:

```

7399 \setglossarystyle{super4col}%

```

Put the glossary in a **supertabular** environment with four columns and a horizontal line in the head and tail:

```

7400 \renewenvironment{theglossary}%
7401 {\tablehead{\hline}\tabletail{\hline}%
7402 \begin{supertabular}{|l|l|l|l|}%
7403 {\end{supertabular}}}%
7404 }

```

per4colheaderborder The **super4colheaderborder** style is like the **super4col** but with a header and border.

```

7405 \newglossarystyle{super4colheaderborder}{%

```

Base it on the **glostylesuper4col** style:

```

7406 \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:

```

7407 \renewenvironment{theglossary}%
7408 {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
7409 \bfseries\symbolname &
7410 \bfseries\pagelistname\tabularnewline\hline}%
7411 \tabletail{\hline}%
7412 \begin{supertabular}{|l|l|l|l|}%
7413 {\end{supertabular}}}%
7414 }

```

altsuper4col The **altsuper4col** glossary style is like **super4col** but has provision for multiline descriptions.

```

7415 \newglossarystyle{altsuper4col}{%

```

Base it on the **glostylesuper4col** style:

```

7416 \setglossarystyle{super4col}%

```

Put the glossary in a **supertabular** environment with four columns and no head or tail:

```

7417 \renewenvironment{theglossary}%
7418 {\tablehead{}\tabletail{}}%
7419 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7420 {\end{supertabular}}}%
7421 }

```

altsuper4colheader The altsuper4colheader style is like the altsuper4col but with a header row.

```

7422 \newglossarystyle{altsuper4colheader}{%
    Base it on the glostylesuper4colheader style:
7423   \setglossarystyle{super4colheader}%

    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
7424   \renewenvironment{theglossary}%
7425     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7426       \bfseries\symbolname &
7427       \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7428     \begin{supertabular}{\lp{\glsgdescwidth}\lp{\glspagelistwidth}}}%
7429     {\end{supertabular}}}%
7430 }
```

altsuper4colborder The altsuper4colborder style is like the altsuper4col but with a border.

```

7431 \newglossarystyle{altsuper4colborder}{%
    Base it on the glostylesuper4colborder style:
7432   \setglossarystyle{super4colborder}%

    Put the glossary in a supertabular environment with four columns and a hori-
    zontal line in the head and tail:
7433   \renewenvironment{theglossary}%
7434     {\tablehead{\hline}\tabletail{\hline}%
7435     \begin{supertabular}%
7436       {\lllp{\glsgdescwidth}\lllp{\glspagelistwidth}}}%
7437     {\end{supertabular}}}%
7438 }
```

per4colheaderborder The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

7439 \newglossarystyle{altsuper4colheaderborder}{%
    Base it on the glostylesuper4colheaderborder style:
7440   \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:
7441   \renewenvironment{theglossary}%
7442     {\tablehead{\hline
7443       \bfseries\entryname &
7444       \bfseries\descriptionname &
7445       \bfseries\symbolname &
7446       \bfseries\pagelistname\tabularnewline\hline}%
7447     \tabletail{\hline}%
7448     \begin{supertabular}%
7449       {\lllp{\glsgdescwidth}\lllp{\glspagelistwidth}}}%
7450     {\end{supertabular}}}%
7451 }
```

4.8 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.

7452 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]

Requires the package:

7453 \RequirePackage{array}

Requires the package:

7454 \RequirePackage{supertabular}

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

7455 \@ifundefined{glsdescwidth}{%

7456 \newlength{glsdescwidth}

7457 \setlength{glsdescwidth}{0.6\hsize}

7458 }{}

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

7459 \@ifundefined{glspagelistwidth}{%

7460 \newlength{glspagelistwidth}

7461 \setlength{glspagelistwidth}{0.1\hsize}

7462 }{}

`superragged` The superragged glossary style uses the supertabular environment.

7463 \newglossarystyle{superragged}{%

Put the glossary in a supertabular environment with two columns and no head or tail:

7464 \renewenvironment{theglossary}{%

7465 {\tablehead{}\tabletail}{%

7466 \begin{supertabular}{1>\raggedright}p{glsdescwidth}}{%

7467 {\end{supertabular}}{%

Do nothing at the start of the table:

7468 \renewcommand*{\glossaryheader}{}{%

No group headings:

7469 \renewcommand*{\glsgroupheading}[1]{}{%

Main (level 0) entries put in a row (name in first column, description and page list in second column):

7470 \renewcommand{\glossentry}[2]{%

7471 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &

7472 \glossentrydesc{##1}\glspostdescription\space ##2%

7473 \tabularnewline

7474 }{%

Sub entries put in a row (no name, description and page list in second column):

```

7475 \renewcommand{\subglossentry}[3]{%
7476     &
7477     \glssubentryitem{##2}%
7478     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7479     ##3%
7480     \tabularnewline
7481 }%
```

Blank row between groups:

```

7482 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
7483 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```

7484 \newglossarystyle{superraggedborder}{%
    Base it on the glostylesuperragged style:
7485 \setglossarystyle{superragged}%
    Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:
7486 \renewenvironment{theglossary}%
7487     {\tablehead{\hline}\tabletail{\hline}%
7488     \begin{supertabular}{|l|>\raggedright}p{\glsdescwidth}|}%
7489     {\end{supertabular}}%
7490 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```

7491 \newglossarystyle{superraggedheader}{%
    Base it on the glostylesuperragged style:
7492 \setglossarystyle{superragged}%
    Put the glossary in a supertabular environment with two columns, a header and no tail:
7493 \renewenvironment{theglossary}%
7494     {\tablehead{\bfseries \entryname & \bfseries \descriptionname
7495         \tabularnewline}%
7496     \tabletail{}%
7497     \begin{supertabular}{l>\raggedright}p{\glsdescwidth}}}%
7498     {\end{supertabular}}%
7499 }
```

rraggedheaderborder The superraggedheaderborder style is like the superragged style but with a header and border:

```

7500 \newglossarystyle{superraggedheaderborder}{%
    Base it on the glostylesuper style:
7501 \setglossarystyle{superragged}%
```


Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

7502 \renewenvironment{theglossary}%
7503   {\tablehead{\hline\bfseries \entryname &
7504     \bfseries \descriptionname\tabularnewline\hline}%
7505     \tabletail{\hline}
7506     \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
7507     {\end{supertabular}}%
7508 }

```

superragged3col The superragged3col style is like the superragged style, but with 3 columns:

```

7509 \newglossarystyle{superragged3col}{%

```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

7510 \renewenvironment{theglossary}%
7511   {\tablehead{}\tabletail{}%
7512     \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%
7513       >{\raggedright}p{\glspagelistwidth}|}}%
7514     {\end{supertabular}}%

```

Do nothing at the start of the table:

```

7515 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

7516 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7517 \renewcommand{\glossentry}[2]{%
7518   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7519   \glossentrydesc{##1} &
7520   ##2\tabularnewline
7521 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

7522 \renewcommand{\subglossentry}[3]{%
7523   &
7524   \glssubentryitem{##2}%
7525   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7526   ##3\tabularnewline
7527 }%

```

Blank row between groups:

```

7528 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &\tabularnewline\fi}%
7529 }

```

superragged3colborder The superragged3colborder style is like the superragged3col style, but with a border:

```

7530 \newglossarystyle{superragged3colborder}{%

```

Base it on the `glostylesuperragged3col` style:

```
7531 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```
7532 \renewenvironment{theglossary}%  
7533   {\tablehead{\hline}\tabletail{\hline}}%  
7534   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%  
7535     >{\raggedright}p{\glspagelistwidth}|}%  
7536   {\end{supertabular}}%  
7537 }
```

`superragged3colheader` The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```
7538 \newglossarystyle{superragged3colheader}{%
```

Base it on the `glostylesuperragged3col` style:

```
7539 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```
7540 \renewenvironment{theglossary}%  
7541   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
7542     \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
7543   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}%  
7544     >{\raggedright}p{\glspagelistwidth}}}%  
7545   {\end{supertabular}}%  
7546 }
```

`superragged3colheaderborder` The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```
7547 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the `glostylesuperragged3colborder` style:

```
7548 \setglossarystyle{superragged3colborder}%
```

Put the glossary in a `supertabular` environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
7549 \renewenvironment{theglossary}%  
7550   {\tablehead{\hline  
7551     \bfseries\entryname&\bfseries\descriptionname&  
7552     \bfseries\pagelistname\tabularnewline\hline}%  
7553   \tabletail{\hline}}%  
7554   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%  
7555     >{\raggedright}p{\glspagelistwidth}|}%  
7556   {\end{supertabular}}%  
7557 }
```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```
7558 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
7559 \renewenvironment{theglossary}%
7560   {\tablehead{}\tabletail{}}%
7561   \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}l%
7562     >{\raggedright}p{\glspagelistwidth}}}%
7563   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7564 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7565 \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:

```
7566 \renewcommand{\glossentry}[2]{%
7567   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7568   \glossentrydesc{##1} &
7569   \glossentrysymbol{##1} & ##2\tabularnewline
7570 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
7571 \renewcommand{\subglossentry}[3]{%
7572   &
7573   \glssubentryitem{##2}%
7574   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7575   \glossentrysymbol{##2} & ##3\tabularnewline
7576 }%
```

Blank row between groups:

```
7577 \renewcommand*{\glsgroupskip}{\ifglsgnogroupskip\else & & \tabularnewline\fi}%
7578 }
```

`\perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```
7579 \newglossarystyle{altsuperragged4colheader}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
7580 \setglossarystyle{altsuperragged4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
7581 \renewenvironment{theglossary}%
7582   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7583     \bfseries\symbolname &
7584     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7585   \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}l%
7586     >{\raggedright}p{\glspagelistwidth}}}%
7587   {\end{supertabular}}%
7588 }
```

`altsuperragged4colborder` The `altsuperragged4colborder` style is like the `altsuperragged4col` style but with a border.

```
7589 \newglossarystyle{altsuperragged4colborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
7590 \setglossarystyle{altsuper4col}{%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
7591 \renewenvironment{theglossary}{%
7592   {\tablehead{\hline}\tabletail{\hline}}%
7593   \begin{supertabular}%
7594     {\l|>{\raggedright}p{\glstdescwidth}|l|}%
7595     >{\raggedright}p{\glspagelistwidth}|}}%
7596   {\end{supertabular}}}%
7597 }
```

`altd4colheaderborder` The `altd4colheaderborder` style is like the `altd4col` style but with a header and border.

```
7598 \newglossarystyle{altd4colheaderborder}{%
```

Base it on the `glostylealtd4col` style:

```
7599 \setglossarystyle{altd4col}{%
```

Put the glossary in a `supertabular` environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```
7600 \renewenvironment{theglossary}{%
7601   {\tablehead{\hline
7602     \bfseries\entryname &
7603     \bfseries\descriptionname &
7604     \bfseries\symbolname &
7605     \bfseries\pagelistname\tablearnewline\hline}%
7606   \tabletail{\hline}%
7607   \begin{supertabular}%
7608     {\l|>{\raggedright}p{\glstdescwidth}|l|}%
7609     >{\raggedright}p{\glspagelistwidth}|}}%
7610   {\end{supertabular}}}%
7611 }
```

4.9 Tree Styles (`glossary-tree.sty`)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

```
7612 \ProvidesPackage{glossary-tree}[2013/11/14 v4.0 (NLCT)]
```

`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.

```
7613 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by theindex:

```
7614 \renewenvironment{theglossary}%
7615   {\setlength{\parindent}{0pt}%
7616    \setlength{\parskip}{0pt plus 0.3pt}%
7617    \let\item\@idxitem}%

7618   {\par}%
```

Do nothing at the start of the environment:

```
7619 \renewcommand*{\glossaryheader}{}%
```

No group headers:

```
7620 \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.

```
7621 \renewcommand*{\glossentry}[2]{%
7622   \item\glsentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7623   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7624   \space \glossentrydesc{##1}\glspostdescription\space ##2%
7625 }%
```

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.

```
7626 \renewcommand{\subglossentry}[3]{%
7627   \ifcase##1\relax
7628     % level 0
7629     \item
7630   \or
7631     % level 1
7632     \subitem
7633     \glssubentryitem{##2}%
7634   \else
7635     % all other levels
7636     \subsubitem
7637   \fi
7638   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7639   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7640   \space\glossentrydesc{##2}\glspostdescription\space ##3%
7641 }%
```

Vertical gap between groups is the same as that used by indices:

```
7642 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

indexgroup The `indexgroup` style is like the `index` style but has headings.

```
7643 \newglossarystyle{indexgroup}{}%
```

Base it on the `glostyleindex` style:

```
7644 \setglossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```
7645 \renewcommand*{\glsgroupheading}[1]{%
7646   \item\textbf{\glsgrouptitle{##1}}\indexspace}%
7647 }
```

`indexhypergroup` The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```
7648 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```
7649 \setglossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```
7650 \renewcommand*{\glossaryheader}{%
7651   \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.

```
7652 \renewcommand*{\glsgroupheading}[1]{%
7653   \item\textbf{\glsnavigationhypertarget{##1}}{\glsgrouptitle{##1}}}%
7654   \indexspace}%
7655 }
```

`tree` The `tree` glossary style is similar in style to the `index` style, but can have arbitrary levels.

```
7656 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
7657 \renewenvironment{theglossary}%
7658   {\setlength{\parindent}{0pt}%
7659    \setlength{\parskip}{0pt plus 0.3pt}}%
7660   {}%
```

Do nothing at the start of the `theglossary` environment:

```
7661 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7662 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
7663 \renewcommand{\glossentry}[2]{%
7664   \hangindent0pt\relax
7665   \parindent0pt\relax
7666   \glstryitem{##1}\textbf{\glstarget{##1}}{\glossentryname{##1}}}%
7667   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7668   \space\glossentrydesc{##1}\glspostdescription\space##2\par
7669 }
```

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.

```

7670 \renewcommand{\subglossentry}[3]{%
7671   \hangindent##1\glstreeindent\relax
7672   \parindent##1\glstreeindent\relax
7673   \ifnum##1=1\relax
7674     \glssubentryitem{##2}%
7675   \fi
7676   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7677   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7678   \space\glossentrydesc{##2}\glspostdescription\space ##3\par
7679 }%
```

Vertical gap between groups is the same as that used by indices:

```

7680 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

treegroup Like the tree style but the glossary groups have headings.

```

7681 \newglossarystyle{treegroup}{%
```

Base it on the `glostyletree` style:

```

7682 \setglossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```

7683 \renewcommand{\glsgroupheading}[1]{\par
7684   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7685 }
```

treehypergroup The `treehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```

7686 \newglossarystyle{treehypergroup}{%
```

Base it on the `glostyletree` style:

```

7687 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```

7688 \renewcommand*{\glossaryheader}{%
7689   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

7690 \renewcommand*{\glsgroupheading}[1]{%
7691   \par\noindent
7692   \textbf{\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}\par
7693   \indexspace}%
7694 }
```

\glstreeindent Length governing left indent for each level of the tree style.

```

7695 \newlength\glstreeindent
7696 \setlength{\glstreeindent}{10pt}
```

treenoname The **treenoname** glossary style is like the **tree** style, but doesn't print the name or symbol for sub-levels.

```
7697 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
7698 \renewenvironment{theglossary}%  
7699 {\setlength{\parindent}{0pt}}%  
7700 \setlength{\parskip}{0pt plus 0.3pt}}%  
7701 {}%
```

No header:

```
7702 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7703 \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.

```
7704 \renewcommand{\glossentry}[2]{%  
7705 \hangindent0pt\relax  
7706 \parindent0pt\relax  
7707 \glstryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%  
7708 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%  
7709 \space\glossentrydesc{##1}\glspostdescription\space##2\par  
7710 }%
```

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.

```
7711 \renewcommand{\subglossentry}[3]{%  
7712 \hangindent##1\glstreeindent\relax  
7713 \parindent##1\glstreeindent\relax  
7714 \ifnum##1=1\relax  
7715 \glssubentryitem{##2}%  
7716 \fi  
7717 \glstarget{##2}{\strut}%  
7718 \glossentrydesc{##2}\glspostdescription\space##3\par  
7719 }%
```

Vertical gap between groups is the same as that used by indices:

```
7720 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%  
7721 }
```

treenonamegroup Like the **treenoname** style but the glossary groups have headings.

```
7722 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
7723 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
7724 \renewcommand{\glsgroupheading}[1]{\par  
7725 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%  
7726 }
```


`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7727 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
7728 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7729 \renewcommand*{\glossaryheader}{%
```

```
7730 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7731 \renewcommand*{\glsgroupheading}[1]{%
```

```
7732 \par\noindent
```

```
7733 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
```

```
7734 \indexspace}%
```

```
7735 }
```

`\glsssetwidest` `\glsssetwidest[<level>]{<text>}` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```
7736 \newcommand*{\glsssetwidest}[2][0]{%
```

```
7737 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
7738 #2}%
```

```
7739 }
```

`\@glswidestname` Initialise `\@glswidestname`.

```
7740 \newcommand*{\@glswidestname}{}
```

`alttree` The `alttree` 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`.

```
7741 \newglossarystyle{alttree}{%
```

Redefine the `theglossary` environment.

```
7742 \renewenvironment{theglossary}%
```

```
7743 {\def\@gls@prevlevel{-1}%
```

```
7744 \mbox{}\par}%
```

```
7745 {\par}%
```

Set the header and group headers to nothing.

```
7746 \renewcommand*{\glossaryheader}{}%
```

```
7747 \renewcommand*{\glsgroupheading}[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
7748 \renewcommand{\glossentry}[2]{%
```

If the level hasn't changed, keep the same settings, otherwise change `\glstreeindent` accordingly.

```
7749 \ifnum\@gls@prevlevel=0\relax
```

```
7750 \else
```

Find out how big the indentation should be by measuring the widest entry.

```

7751      \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%

```

Set the hangindent and paragraph indent.

```

7752      \hangindent\glstreeindent
7753      \parindent\glstreeindent
7754      \fi

```

Put the name to the left of the paragraph block.

```

7755      \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
7756        \glssentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

7757      \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%

```

Do the description followed by the description terminator and location list.

```

7758      \glossentrydesc{##1}\glspostdescription \space ##2\par

```

Set the previous level to 0.

```

7759      \def\@gls@prevlevel{0}%
7760      }%

```

Redefine the way sub-entries are displayed.

```

7761      \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.

```

7762      \ifnum##1=1\relax
7763      \glssubentryitem{##2}%
7764      \fi

```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```

7765      \ifnum\@gls@prevlevel=##1\relax
7766      \else

```

Compute the widest entry for this level, or for level 0 if not defined for this level. Store in \gls@tmplen

```

7767      \@ifundefined{@glswidestname\romannumeral##1}{%
7768      \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
7769      \settowidth{\gls@tmplen}{\textbf{%
7770      \csname @glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```

7771      \ifnum\@gls@prevlevel<##1\relax

```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```

7772      \setlength\glstreeindent\gls@tmplen
7773      \addtolength\glstreeindent\parindent
7774      \parindent\glstreeindent
7775      \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`.

```

7776      \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
7777      \settowidth{\glstreeindent}{\textbf{%
7778      \@glswidestname\space}}}{%
7779      \settowidth{\glstreeindent}{\textbf{%
7780      \csname @glswidestname\romannumeral\@gls@prevlevel
7781      \endcsname\space}}}{%

```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```

7782      \addtolength\parindent{-\glstreeindent}%
7783      \setlength\glstreeindent\parindent
7784      \fi
7785      \fi

```

Set the hanging indentation.

```

7786      \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

7787      \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
7788      \textbf{\glstarget{##2}{\glossentryname{##2}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

7789      \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%

```

Do the description followed by the description terminator and location list.

```

7790      \glossentrydesc{##2}\glspostdescription\space ##3\par

```

Set the previous level macro to the current level.

```

7791      \def\@gls@prevlevel{##1}%
7792      }%

```

Vertical gap between groups is the same as that used by indices:

```

7793      \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7794      }

```

almtreegroup Like the almtree style but the glossary groups have headings.

```

7795 \newglossarystyle{almtreegroup}{%

```

Base it on the glostylealmtree style:

```

7796 \setglossarystyle{almtree}%

```

Give each group a heading.

```

7797 \renewcommand{\glsgroupheading}[1]{\par
7798 \def\@gls@prevlevel{-1}%
7799 \hangindent0pt\relax
7800 \parindent0pt\relax
7801 \textbf{\glsgrouptitle{##1}}\par\indexspace}%
7802 }

```

`almtreehypergroup` The `almtreehypergroup` style is like the `almtreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7803 \newglossarystyle{almtreehypergroup}{%
```

Base it on the `glostylealmtree` style:

```
7804 \setglossarystyle{almtree}%
```

Put the navigation links in the header

```
7805 \renewcommand*{\glossaryheader}{%
```

```
7806 \par
```

```
7807 \def\@gls@prevlevel{-1}%
```

```
7808 \hangindent0pt\relax
```

```
7809 \parindent0pt\relax
```

```
7810 \textbf{\glsnavigation}\par\indexspace}%
```

Put a `hypertarget` at the start of each group

```
7811 \renewcommand*{\glsgroupheading}[1]{%
```

```
7812 \par
```

```
7813 \def\@gls@prevlevel{-1}%
```

```
7814 \hangindent0pt\relax
```

```
7815 \parindent0pt\relax
```

```
7816 \textbf{\glsnavhypertarget{##1}\glsgetgrouptitle{##1}}\par
```

```
7817 \indexspace}}
```

5 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.

```
7818 \NeedsTeXFormat{LaTeX2e}
```

```
7819 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]
```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```
7820 \ifglsxindy
```

```
7821 \renewcommand*\GlsAddXdyAttribute[1]{%
```

```
7822 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
```

```
7823 \expandafter\toks@\expandafter{\@xdylocref}%
```

```
7824 \edef\@xdylocref{\the\toks@ ^^J%
```

```
7825 (markup-locref
```

```
7826 :open \string"\string~n\string\setentrycounter
```

```
7827 {\noexpand\glscounter}%
```

```
7828 \expandafter\string\csname#1\endcsname
```

```
7829 \expandafter\@gobble\string\{\string" ^^J
```

```
7830 :close \string"\expandafter\@gobble\string\}\string" ^^J
```

```
7831 :attr \string"#1\string"))}}
```

Only has an effect before `\writeist`:

```
7832 \fi
```

\GlsAddXdyCounters

```
7833 \renewcommand*\GlsAddXdyCounters[1]{%
7834   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
7835     in compatibility mode.}%
7836 }
```

Add predefined attributes

```
7837 \GlsAddXdyAttribute{glsnumberformat}
7838 \GlsAddXdyAttribute{textrm}
7839 \GlsAddXdyAttribute{textsf}
7840 \GlsAddXdyAttribute{texttt}
7841 \GlsAddXdyAttribute{textbf}
7842 \GlsAddXdyAttribute{textmd}
7843 \GlsAddXdyAttribute{textit}
7844 \GlsAddXdyAttribute{textup}
7845 \GlsAddXdyAttribute{textsl}
7846 \GlsAddXdyAttribute{textsc}
7847 \GlsAddXdyAttribute{emph}
7848 \GlsAddXdyAttribute{glshypernumber}
7849 \GlsAddXdyAttribute{hyperrm}
7850 \GlsAddXdyAttribute{hypersf}
7851 \GlsAddXdyAttribute{hypertt}
7852 \GlsAddXdyAttribute{hyperbf}
7853 \GlsAddXdyAttribute{hypermd}
7854 \GlsAddXdyAttribute{hyperit}
7855 \GlsAddXdyAttribute{hyperup}
7856 \GlsAddXdyAttribute{hypersl}
7857 \GlsAddXdyAttribute{hypersc}
7858 \GlsAddXdyAttribute{hyperemph}
```

\GlsAddXdyLocation Restore v2.07 definition:

```
7859 \ifglxindy
7860   \renewcommand*\GlsAddXdyLocation[2]{%
7861     \edef\@xdyuserlocationdefs{%
7862       \@xdyuserlocationdefs ^^J%
7863       (define-location-class \string"#1\string"^^J\space\space
7864         \space(#2))
7865     }%
7866     \edef\@xdyuserlocationnames{%
7867       \@xdyuserlocationnames^^J\space\space\space
7868       \string"#1\string"%
7869     }
7870 \fi
```

\@do@wrglossary

```
7871 \renewcommand*\@do@wrglossary[1]{%
  Determine whether to use xindy or makeindex syntax
7872 \ifglxindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```

7873 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
7874 \def\@glo@range{}%
7875 \expandafter\if\@glo@prefix(\relax
7876 \def\@glo@range{:open-range}%
7877 \else
7878 \expandafter\if\@glo@prefix)\relax
7879 \def\@glo@range{:close-range}%
7880 \fi
7881 \fi

```

Get the location and escape any special characters

```

7882 \protected@edef\@glslocref{\theglsentrycounter}%
7883 \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```

7884 \glossary[\csname glo@#1@type\endcsname]{%
7885 (indexentry :tkey (\csname glo@#1@index\endcsname)
7886 :locref \string"\@glslocref\string" %
7887 :attr \string"\@glo@suffix\string" \@glo@range
7888 )
7889 }%
7890 \else

```

Convert the format information into the format required for makeindex

```

7891 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

```

Write to the glossary file using makeindex syntax.

```

7892 \glossary[\csname glo@#1@type\endcsname]{%
7893 \string\glossaryentry{\csname glo@#1@index\endcsname
7894 \@gls@encapchar\@glo@numfmt}\theglsentrycounter}}%
7895 \fi
7896 }

```

\@set@glo@numformat Only had 3 arguments in v2.07

```

7897 \def\@set@glo@numformat#1#2#3{%
7898 \expandafter\@glo@check@mkidxrangechar#3\@nil
7899 \protected@edef#1{%
7900 \@glo@prefix setentrycounter[] {#2}%
7901 \expandafter\string\csname\@glo@suffix\endcsname
7902 }%
7903 \@gls@checkmkidxchars#1%
7904 }

```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```

7905 \ifglsxindy
7906 \def\writeist{%
7907 \openout\glswrite=\istfilename

```

```

7908 \write\glswrite{;; xindy style file created by the glossaries
7909   package in compatible-2.07 mode}%
7910 \write\glswrite{;; for document '\jobname' on
7911   \the\year-\the\month-\the\day}%
7912 \write\glswrite{^^J; required styles^^J}
7913 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
7914   \ifx\@xdystyle\@empty
7915   \else
7916     \protected@write\glswrite{{(require
7917       \string"\@xdystyle.xdy\string")}}%
7918   \fi
7919 }%
7920 \write\glswrite{^^J%
7921   ; list of allowed attributes (number formats)^^J}%
7922 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
7923 \write\glswrite{^^J; user defined alphabets^^J}%
7924 \write\glswrite{\@xdyuseralphabets}%
7925 \write\glswrite{^^J; location class definitions^^J}%
7926 \protected@edef\@gls@roman{\@roman{0}\string"
7927   \string"roman-numbers-lowercase\string" :sep \string"}}%
7928 \@onelevel@sanitize\@gls@roman
7929 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
7930   :sep \string"}%
7931 \@onelevel@sanitize\@tmp
7932 \ifx\@tmp\@gls@roman
7933   \write\glswrite{(define-location-class
7934     \string"roman-page-numbers\string"^^J\space\space\space
7935     (\string"roman-numbers-lowercase\string")
7936     :min-range-length \@glsminrange)}}%
7937 \else
7938   \write\glswrite{(define-location-class
7939     \string"roman-page-numbers\string"^^J\space\space\space
7940     (:sep "\@gls@roman")
7941     :min-range-length \@glsminrange)}}%
7942 \fi
7943 \write\glswrite{(define-location-class
7944   \string"Roman-page-numbers\string"^^J\space\space\space
7945   (\string"roman-numbers-uppercase\string")
7946   :min-range-length \@glsminrange)}}%
7947 \write\glswrite{(define-location-class
7948   \string"arabic-page-numbers\string"^^J\space\space\space
7949   (\string"arabic-numbers\string")
7950   :min-range-length \@glsminrange)}}%
7951 \write\glswrite{(define-location-class
7952   \string"alpha-page-numbers\string"^^J\space\space\space
7953   (\string"alpha\string")
7954   :min-range-length \@glsminrange)}}%
7955 \write\glswrite{(define-location-class
7956   \string"Alpha-page-numbers\string"^^J\space\space\space

```

```

7957      (\string"ALPHA\string")
7958      :min-range-length \@glsmminrange))}%
7959 \write\glswrite{(define-location-class
7960   \string"Appendix-page-numbers\string"^^J\space\space\space
7961   (\string"ALPHA\string"
7962     :sep \string"\@glAlphacompositor\string"
7963     \string"arabic-numbers\string")
7964     :min-range-length \@glsmminrange))}%
7965 \write\glswrite{(define-location-class
7966   \string"arabic-section-numbers\string"^^J\space\space\space
7967   (\string"arabic-numbers\string"
7968     :sep \string"\glscocompositor\string"
7969     \string"arabic-numbers\string")
7970     :min-range-length \@glsmminrange))}%
7971 \write\glswrite{^^J; user defined location classes}%
7972 \write\glswrite{@xdyuserlocationdefs}%
7973 \write\glswrite{^^J; define cross-reference class^^J}%
7974 \write\glswrite{(define-crossref-class \string"see\string"
7975   :unverified )}%
7976 \write\glswrite{(markup-crossref-list
7977   :class \string"see\string"^^J\space\space\space
7978   :open \string"\string\glseeformat\string"
7979   :close \string"{}\string")}%
7980 \write\glswrite{^^J; define the order of the location classes}%
7981 \write\glswrite{(define-location-class-order
7982   (\@xdylocationclassorder))}%
7983 \write\glswrite{^^J; define the glossary markup^^J}%
7984 \write\glswrite{(markup-index^^J\space\space\space
7985   :open \string"\string
7986     \glossarysection[\string\glossarytoctitle]{\string
7987     \glossarytitle}\string\glossarypreamble\string~n\string\begin
7988     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
7989     \space\space:close \string"\expandafter\@gobble
7990     \string\%\string~n\string
7991     \end{theglossary}\string\glossarypostamble
7992     \string~n\string" ^^J\space\space\space
7993     :tree)}}%
7994 \write\glswrite{(markup-letter-group-list
7995   :sep \string"\string\glsgroupskip\string~n\string")}%
7996 \write\glswrite{(markup-indexentry
7997   :open \string"\string\relax \string\glresetentrylist
7998     \string~n\string")}%
7999 \write\glswrite{(markup-locclass-list :open
8000   \string"\glsoopenbrace\string\glossaryentrynumbers
8001   \glsoopenbrace\string\relax\space \string"^^J\space\space\space
8002   :sep \string", \string"
8003   :close \string"\glsclosebrace\glsclosebrace\string")}%
8004 \write\glswrite{(markup-locref-list
8005   :sep \string"\string\delimN\space\string")}%

```



```

8006 \write\glswrite{(markup-range
8007 :sep \string"\string\delimR\space\string")}%
8008 \@onelevel@sanitize\gls@suffixF
8009 \@onelevel@sanitize\gls@suffixFF
8010 \ifx\gls@suffixF\@empty
8011 \else
8012 \write\glswrite{(markup-range
8013 :close "\gls@suffixF" :length 1 :ignore-end)}%
8014 \fi
8015 \ifx\gls@suffixFF\@empty
8016 \else
8017 \write\glswrite{(markup-range
8018 :close "\gls@suffixFF" :length 2 :ignore-end)}%
8019 \fi
8020 \write\glswrite{^^J; define format to use for locations^^J}%
8021 \write\glswrite{\@xdylocref}%
8022 \write\glswrite{^^J; define letter group list format^^J}%
8023 \write\glswrite{(markup-letter-group-list
8024 :sep \string"\string\glsgroupskip\string~\n\string")}%
8025 \write\glswrite{^^J; letter group headings^^J}%
8026 \write\glswrite{(markup-letter-group
8027 :open-head \string"\string\glsgroupheading
8028 \glsopenbrace\string"^^J\space\space\space
8029 :close-head \string"\glsclosebrace\string")}%
8030 \write\glswrite{^^J; additional letter groups^^J}%
8031 \write\glswrite{\@xdylettergroups}%
8032 \write\glswrite{^^J; additional sort rules^^J}%
8033 \write\glswrite{\@xdysortrules}%
8034 \noist}
8035 \else
8036 \edef\@gls@actualchar{\string?}
8037 \edef\@gls@encapchar{\string|}
8038 \edef\@gls@levelchar{\string!}
8039 \edef\@gls@quotechar{\string"}
8040 \def\writeist{\relax
8041 \openout\glswrite=\istfilename
8042 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8043 created by the glossaries package}
8044 \write\glswrite{\expandafter\@gobble\string\% for document
8045 '\jobname' on \the\year-\the\month-\the\day}
8046 \write\glswrite{actual '\@gls@actualchar'}
8047 \write\glswrite{encap '\@gls@encapchar'}
8048 \write\glswrite{level '\@gls@levelchar'}
8049 \write\glswrite{quote '\@gls@quotechar'}
8050 \write\glswrite{keyword \string"\string\glossaryentry\string"}
8051 \write\glswrite{preamble \string"\string\glossarysection[\string
8052 \glossarytoctitle]{\string\glossarytitle}\string
8053 \glossarypreamble\string\n\string\begin{theglossary}\string
8054 \glossaryheader\string\n\string"}

```

```

8055 \write\glswrite{postamble \string"\string%\string\n\string
8056 \end{theglossary}\string\glossarypostamble\string\n
8057 \string"}
8058 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8059 \string"}
8060 \write\glswrite{item_0 \string"\string%\string\n\string"}
8061 \write\glswrite{item_1 \string"\string%\string\n\string"}
8062 \write\glswrite{item_2 \string"\string%\string\n\string"}
8063 \write\glswrite{item_01 \string"\string%\string\n\string"}
8064 \write\glswrite{item_x1
8065 \string"\string\relax \string\glresetentrylist\string\n
8066 \string"}
8067 \write\glswrite{item_12 \string"\string%\string\n\string"}
8068 \write\glswrite{item_x2
8069 \string"\string\relax \string\glresetentrylist\string\n
8070 \string"}
8071 \write\glswrite{delim_0 \string"\string\{\string
8072 \glossaryentrynumbers\string\{\string\relax \string"}
8073 \write\glswrite{delim_1 \string"\string\{\string
8074 \glossaryentrynumbers\string\{\string\relax \string"}
8075 \write\glswrite{delim_2 \string"\string\{\string
8076 \glossaryentrynumbers\string\{\string\relax \string"}
8077 \write\glswrite{delim_t \string"\string\}\string\}\string"}
8078 \write\glswrite{delim_n \string"\string\delimN \string"}
8079 \write\glswrite{delim_r \string"\string\delimR \string"}
8080 \write\glswrite{headings_flag 1}
8081 \write\glswrite{heading_prefix
8082 \string"\string\glsgroupheading\string\{\string"}
8083 \write\glswrite{heading_suffix
8084 \string"\string\}\string\relax
8085 \string\glresetentrylist \string"}
8086 \write\glswrite{symhead_positive \string"glssymbols\string"}
8087 \write\glswrite{numhead_positive \string"glnumbers\string"}
8088 \write\glswrite{page_compositor \string"glcompositor\string"}
8089 \@glscbsdq\glscsuffixF
8090 \@glscbsdq\glscsuffixFF
8091 \ifx\glscsuffixF\empty
8092 \else
8093 \write\glswrite{suffix_2p \string"\glscsuffixF\string"}
8094 \fi
8095 \ifx\glscsuffixFF\empty
8096 \else
8097 \write\glswrite{suffix_3p \string"\glscsuffixFF\string"}
8098 \fi
8099 \noist
8100 }
8101 \fi

```

\noist

```
8102 \renewcommand*{\noist}{\let\writeist\relax}
```

Compatibility macros.

```
8103 \NeedsTeXFormat{LaTeX2e}
```

```
8104 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]
```

Compatibility macros for predefined glossary styles:

`compatglossarystyle` Defines a compatibility glossary style.

```
8105 \newcommand{\compatglossarystyle}[2]{%
8106   \ifcsundef{@glscompstyle@#1}%
8107   {%
8108     \csdef{@glscompstyle@#1}{#2}%
8109   }%
8110   {%
8111     \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8112   }%
8113 }
```

Backward compatible inline style.

```
8114 \compatglossarystyle{inline}{%
8115   \renewcommand{\glossaryentryfield}[5]{%
8116     \glsinlinedopostchild
8117     \gls@inlinesep
8118     \def\glo@desc{##3}%
8119     \def\@no@post@desc{\nopostdesc}%
8120     \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8121     \ifx\glo@desc\@no@post@desc
8122       \glsinlineemptydescformat{##4}{##5}%
8123     \else
8124       \ifstrepty{##3}%
8125         {\glsinlineemptydescformat{##4}{##5}}%
8126         {\glsinlinedescformat{##3}{##4}{##5}}%
8127       \fi
8128       \ifglshaschildren{##1}%
8129       {%
8130         \glsresetsubentrycounter
8131         \glsinlineparentchildseparator
8132         \def\gls@inlinesubsep{}%
8133         \def\gls@inlinepostchild{\glsinlinepostchild}%
8134       }%
8135     }%
8136     \def\gls@inlinesep{\glsinlineseparator}%
8137   }%
```

Sub-entries display description:

```
8138 \renewcommand{\glossarysubentryfield}[6]{%
8139   \gls@inlinesubsep%
8140   \glsinlinesubnameformat{##2}{##3}%
8141   \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
```

```

8142 \def\gls@inlinesubsep{\glsinlinesubseparator}%
8143 }%
8144 }

```

Backward compatible list style.

```

8145 \compatglossarystyle{list}{%
8146 \renewcommand*{\glossaryentryfield}[5]{%
8147 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8148 ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

8149 \renewcommand*{\glossarysubentryfield}[6]{%
8150 \glssubentryitem{##2}%
8151 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8152 }

```

Backward compatible listgroup style.

```

8153 \compatglossarystyle{listgroup}{%
8154 \csuse{@glscompstyle@list}%
8155 }%

```

Backward compatible listhypergroup style.

```

8156 \compatglossarystyle{listhypergroup}{%
8157 \csuse{@glscompstyle@list}%
8158 }%

```

Backward compatible altlist style.

```

8159 \compatglossarystyle{altlist}{%
8160 \renewcommand*{\glossaryentryfield}[5]{%
8161 \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
8162 \mbox{\par\nobreak\@afterheading
8163 ##3\glspostdescription\space ##5}%
8164 \renewcommand{\glossarysubentryfield}[6]{%
8165 \par
8166 \glssubentryitem{##2}%
8167 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8168 }%

```

Backward compatible altlistgroup style.

```

8169 \compatglossarystyle{altlistgroup}{%
8170 \csuse{@glscompstyle@altlist}%
8171 }%

```

Backward compatible altlisthypergroup style.

```

8172 \compatglossarystyle{altlisthypergroup}{%
8173 \csuse{@glscompstyle@altlist}%
8174 }%

```

Backward compatible listdotted style.

```

8175 \compatglossarystyle{listdotted}{%
8176 \renewcommand*{\glossaryentryfield}[5]{%
8177 \item[\makebox[\glslistdottedwidth][l]{%

```

```

8178      \glstryitem{##1}\glstarget{##1}{##2}%
8179      \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
8180 \renewcommand*{\glossarysubentryfield}[6]{%
8181   \item[]\makebox[\glslstdottedwidth][l]{%
8182     \glssubentryitem{##2}%
8183     \glstarget{##2}{##3}%
8184     \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
8185 }%

```

Backward compatible sublistdotted style.

```

8186 \compatglossarystyle{sublistdotted}{%
8187   \csuse{@glscmpstyle@listdotted}%
8188   \renewcommand*{\glossaryentryfield}[5]{%
8189     \item[\glstryitem{##1}\glstarget{##1}{##2}]}%
8190 }%

```

Backward compatible long style.

```

8191 \compatglossarystyle{long}{%
8192   \renewcommand*{\glossaryentryfield}[5]{%
8193     \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8194   \renewcommand*{\glossarysubentryfield}[6]{%
8195     &
8196     \glssubentryitem{##2}%
8197     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8198 }%

```

Backward compatible longborder style.

```

8199 \compatglossarystyle{longborder}{%
8200   \csuse{@glscmpstyle@long}%
8201 }%

```

Backward compatible longheader style.

```

8202 \compatglossarystyle{longheader}{%
8203   \csuse{@glscmpstyle@long}%
8204 }%

```

Backward compatible longheaderborder style.

```

8205 \compatglossarystyle{longheaderborder}{%
8206   \csuse{@glscmpstyle@long}%
8207 }%

```

Backward compatible long3col style.

```

8208 \compatglossarystyle{long3col}{%
8209   \renewcommand*{\glossaryentryfield}[5]{%
8210     \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8211   \renewcommand*{\glossarysubentryfield}[6]{%
8212     &
8213     \glssubentryitem{##2}%
8214     \glstarget{##2}{\strut}##4 & ##6\\}%
8215 }%

```

Backward compatible long3colborder style.

```

8216 \compatglossarystyle{long3colborder}{%
8217  \csuse{@glscompstyle@long3col}%
8218 }%

```

Backward compatible long3colheader style.

```

8219 \compatglossarystyle{long3colheader}{%
8220  \csuse{@glscompstyle@long3col}%
8221 }%

```

Backward compatible long3colheaderborder style.

```

8222 \compatglossarystyle{long3colheaderborder}{%
8223  \csuse{@glscompstyle@long3col}%
8224 }%

```

Backward compatible long4col style.

```

8225 \compatglossarystyle{long4col}{%
8226  \renewcommand*{\glossaryentryfield}[5]{%
8227    \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8228  \renewcommand*{\glossarysubentryfield}[6]{%
8229    &
8230    \glssubentryitem{##2}%
8231    \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8232 }%

```

Backward compatible long4colheader style.

```

8233 \compatglossarystyle{long4colheader}{%
8234  \csuse{@glscompstyle@long4col}%
8235 }%

```

Backward compatible long4colborder style.

```

8236 \compatglossarystyle{long4colborder}{%
8237  \csuse{@glscompstyle@long4col}%
8238 }%

```

Backward compatible long4colheaderborder style.

```

8239 \compatglossarystyle{long4colheaderborder}{%
8240  \csuse{@glscompstyle@long4col}%
8241 }%

```

Backward compatible altlong4col style.

```

8242 \compatglossarystyle{altlong4col}{%
8243  \csuse{@glscompstyle@long4col}%
8244 }%

```

Backward compatible altlong4colheader style.

```

8245 \compatglossarystyle{altlong4colheader}{%
8246  \csuse{@glscompstyle@long4col}%
8247 }%

```

Backward compatible altlong4colborder style.

```

8248 \compatglossarystyle{altlong4colborder}{%
8249  \csuse{@glscompstyle@long4col}%
8250 }%

```

Backward compatible altlong4colheaderborder style.

```
8251 \compatglossarystyle{altlong4colheaderborder}{%  
8252 \csuse{@glscmpstyle@long4col}%  
8253 }%
```

Backward compatible long style.

```
8254 \compatglossarystyle{longragged}{%  
8255 \renewcommand*{\glossaryentryfield}[5]{%  
8256 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
8257 \tabularnewline}%  
8258 \renewcommand*{\glossarysubentryfield}[6]{%  
8259 &  
8260 \glssubentryitem{##2}%  
8261 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
8262 \tabularnewline}%  
8263 }%
```

Backward compatible longraggedborder style.

```
8264 \compatglossarystyle{longraggedborder}{%  
8265 \csuse{@glscmpstyle@longragged}%  
8266 }%
```

Backward compatible longraggedheader style.

```
8267 \compatglossarystyle{longraggedheader}{%  
8268 \csuse{@glscmpstyle@longragged}%  
8269 }%
```

Backward compatible longraggedheaderborder style.

```
8270 \compatglossarystyle{longraggedheaderborder}{%  
8271 \csuse{@glscmpstyle@longragged}%  
8272 }%
```

Backward compatible longragged3col style.

```
8273 \compatglossarystyle{longragged3col}{%  
8274 \renewcommand*{\glossaryentryfield}[5]{%  
8275 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%  
8276 \renewcommand*{\glossarysubentryfield}[6]{%  
8277 &  
8278 \glssubentryitem{##2}%  
8279 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%  
8280 }%
```

Backward compatible longragged3colborder style.

```
8281 \compatglossarystyle{longragged3colborder}{%  
8282 \csuse{@glscmpstyle@longragged3col}%  
8283 }%
```

Backward compatible longragged3colheader style.

```
8284 \compatglossarystyle{longragged3colheader}{%  
8285 \csuse{@glscmpstyle@longragged3col}%  
8286 }%
```

Backward compatible longragged3colheaderborder style.

```
8287 \compatglossarystyle{longragged3colheaderborder}{%
8288 \csuse{@glscmpstyle@longragged3col}%
8289 }%
```

Backward compatible altlongragged4col style.

```
8290 \compatglossarystyle{altlongragged4col}{%
8291 \renewcommand*{\glossaryentryfield}[5]{%
8292 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8293 \renewcommand*{\glossarysubentryfield}[6]{%
8294 &
8295 \glssubentryitem{##2}%
8296 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8297 }%
```

Backward compatible altlongragged4colheader style.

```
8298 \compatglossarystyle{altlongragged4colheader}{%
8299 \csuse{@glscmpstyle@altlong4col}%
8300 }%
```

Backward compatible altlongragged4colborder style.

```
8301 \compatglossarystyle{altlongragged4colborder}{%
8302 \csuse{@glscmpstyle@altlong4col}%
8303 }%
```

Backward compatible altlongragged4colheaderborder style.

```
8304 \compatglossarystyle{altlongragged4colheaderborder}{%
8305 \csuse{@glscmpstyle@altlong4col}%
8306 }%
```

Backward compatible index style.

```
8307 \compatglossarystyle{index}{%
8308 \renewcommand*{\glossaryentryfield}[5]{%
8309 \item\glstryitem{##1}\textbf{\glstarget{##1}{##2}}%
8310 \ifx\relax##4\relax
8311 \else
8312 \space{##4}%
8313 \fi
8314 \space ##3\glspostdescription \space ##5}%
8315 \renewcommand*{\glossarysubentryfield}[6]{%
8316 \ifcase##1\relax
8317 % level 0
8318 \item
8319 \or
8320 % level 1
8321 \subitem
8322 \glssubentryitem{##2}%
8323 \else
8324 % all other levels
8325 \subsubitem
8326 \fi
```



```

8327 \textbf{\glstarget{##2}{##3}}%
8328 \ifx\relax##5\relax
8329 \else
8330 \space(##5)%
8331 \fi
8332 \space##4\glspostdescription\space ##6}%
8333 }%

```

Backward compatible indexgroup style.

```

8334 \compatglossarystyle{indexgroup}{%
8335 \csuse{@glscmpstyle@index}%
8336 }%

```

Backward compatible indexhypergroup style.

```

8337 \compatglossarystyle{indexhypergroup}{%
8338 \csuse{@glscmpstyle@index}%
8339 }%

```

Backward compatible tree style.

```

8340 \compatglossarystyle{tree}{%
8341 \renewcommand{\glossaryentryfield}[5]{%
8342 \hangindent0pt\relax
8343 \parindent0pt\relax
8344 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8345 \ifx\relax##4\relax
8346 \else
8347 \space(##4)%
8348 \fi
8349 \space ##3\glspostdescription \space ##5\par}%
8350 \renewcommand{\glossarysubentryfield}[6]{%
8351 \hangindent##1\glstreeindent\relax
8352 \parindent##1\glstreeindent\relax
8353 \ifnum##1=1\relax
8354 \glssubentryitem{##2}%
8355 \fi
8356 \textbf{\glstarget{##2}{##3}}%
8357 \ifx\relax##5\relax
8358 \else
8359 \space(##5)%
8360 \fi
8361 \space##4\glspostdescription\space ##6\par}%
8362 }%

```

Backward compatible treegroup style.

```

8363 \compatglossarystyle{treegroup}{%
8364 \csuse{@glscmpstyle@tree}%
8365 }%

```

Backward compatible treehypergroup style.

```

8366 \compatglossarystyle{treehypergroup}{%
8367 \csuse{@glscmpstyle@tree}%
8368 }%

```

Backward compatible treenoname style.

```

8369 \compatglossarystyle{treenoname}{%
8370   \renewcommand{\glossaryentryfield}[5]{%
8371     \hangindent0pt\relax
8372     \parindent0pt\relax
8373     \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8374     \ifx\relax##4\relax
8375       \else
8376         \space{##4}%
8377       \fi
8378       \space ##3\glspostdescription \space ##5\par}%
8379   \renewcommand{\glossarysubentryfield}[6]{%
8380     \hangindent##1\glstreeindent\relax
8381     \parindent##1\glstreeindent\relax
8382     \ifnum##1=1\relax
8383       \glssubentryitem{##2}%
8384     \fi
8385     \glstarget{##2}{\strut}%
8386     ##4\glspostdescription\space ##6\par}%
8387 }%
```

Backward compatible treenonamegroup style.

```

8388 \compatglossarystyle{treenonamegroup}{%
8389   \csuse{@glscmpstyle@treenoname}%
8390 }%
```

Backward compatible treenonamehypergroup style.

```

8391 \compatglossarystyle{treenonamehypergroup}{%
8392   \csuse{@glscmpstyle@treenoname}%
8393 }%
```

Backward compatible alttree style.

```

8394 \compatglossarystyle{alttree}{%
8395   \renewcommand{\glossaryentryfield}[5]{%
8396     \ifnum\@gls@prevlevel=0\relax
8397     \else
8398       \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
8399       \hangindent\glstreeindent
8400       \parindent\glstreeindent
8401     \fi
8402     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
8403       \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}}%
8404     \ifx\relax##4\relax
8405     \else
8406       (##4)\space
8407     \fi
8408     ##3\glspostdescription \space ##5\par
8409     \def\@gls@prevlevel{0}%
8410   }%
8411   \renewcommand{\glossarysubentryfield}[6]{%
```

```

8412 \ifnum##1=1\relax
8413 \glssubentryitem{##2}%
8414 \fi
8415 \ifnum\@gls@prevlevel=##1\relax
8416 \else
8417 \@ifundefined{@glswidestname\romannumeral##1}{%
8418 \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
8419 \settowidth{\gls@tmplen}{\textbf{%
8420 \csname @glswidestname\romannumeral##1\endcsname\space}}}%
8421 \ifnum\@gls@prevlevel<##1\relax
8422 \setlength\glstreeindent\gls@tmplen
8423 \addtolength\glstreeindent\parindent
8424 \parindent\glstreeindent
8425 \else
8426 \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
8427 \settowidth{\glstreeindent}{\textbf{%
8428 \@glswidestname\space}}{%
8429 \settowidth{\glstreeindent}{\textbf{%
8430 \csname @glswidestname\romannumeral\@gls@prevlevel
8431 \endcsname\space}}}%
8432 \addtolength\parindent{-\glstreeindent}%
8433 \setlength\glstreeindent\parindent
8434 \fi
8435 \fi
8436 \hangindent\glstreeindent
8437 \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
8438 \textbf{\glstarget{##2}{##3}}}%
8439 \ifx##5\relax\relax
8440 \else
8441 (##5)\space
8442 \fi
8443 ##4\glspostdescription\space ##6\par
8444 \def\@gls@prevlevel{##1}%
8445 }%
8446 }%

```

Backward compatible alttreegroup style.

```

8447 \compatglossarystyle{alttreegroup}{%
8448 \csuse{@glscompstyle@alttree}%
8449 }%

```

Backward compatible alttreehypergroup style.

```

8450 \compatglossarystyle{alttreehypergroup}{%
8451 \csuse{@glscompstyle@alttree}%
8452 }%

```

Backward compatible mcolindex style.

```

8453 \compatglossarystyle{mcolindex}{%
8454 \csuse{@glscompstyle@index}%
8455 }%

```

Backward compatible mcolindexgroup style.

```
8456 \compatglossarystyle{mcolindexgroup}{%  
8457 \csuse{@glscompstyle@index}%  
8458 }%
```

Backward compatible mcolindexhypergroup style.

```
8459 \compatglossarystyle{mcolindexhypergroup}{%  
8460 \csuse{@glscompstyle@index}%  
8461 }%
```

Backward compatible mcoltree style.

```
8462 \compatglossarystyle{mcoltree}{%  
8463 \csuse{@glscompstyle@tree}%  
8464 }%
```

Backward compatible mcoltreegroup style.

```
8465 \compatglossarystyle{mcolindextreegroup}{%  
8466 \csuse{@glscompstyle@tree}%  
8467 }%
```

Backward compatible mcoltreehypergroup style.

```
8468 \compatglossarystyle{mcolindextreehypergroup}{%  
8469 \csuse{@glscompstyle@tree}%  
8470 }%
```

Backward compatible mcoltreenoname style.

```
8471 \compatglossarystyle{mcoltreenoname}{%  
8472 \csuse{@glscompstyle@tree}%  
8473 }%
```

Backward compatible mcoltreenonamegroup style.

```
8474 \compatglossarystyle{mcoltreenonamegroup}{%  
8475 \csuse{@glscompstyle@tree}%  
8476 }%
```

Backward compatible mcoltreenonamehypergroup style.

```
8477 \compatglossarystyle{mcoltreenonamehypergroup}{%  
8478 \csuse{@glscompstyle@tree}%  
8479 }%
```

Backward compatible mcolalttree style.

```
8480 \compatglossarystyle{mcolalttree}{%  
8481 \csuse{@glscompstyle@alttree}%  
8482 }%
```

Backward compatible mcolalttreegroup style.

```
8483 \compatglossarystyle{mcolalttreegroup}{%  
8484 \csuse{@glscompstyle@alttree}%  
8485 }%
```

Backward compatible mcolalttreehypergroup style.

```
8486 \compatglossarystyle{mcolalttreehypergroup}{%  
8487 \csuse{@glscompstyle@alttree}%  
8488 }%
```

Backward compatible superragged style.

```
8489 \compatglossarystyle{superragged}{%
8490   \renewcommand*{\glossaryentryfield}[5]{%
8491     \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
8492     \tabularnewline}%
8493   \renewcommand*{\glossarysubentryfield}[6]{%
8494     &
8495     \glssubentryitem{##2}%
8496     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
8497     \tabularnewline}%
8498 }%
```

Backward compatible superraggedborder style.

```
8499 \compatglossarystyle{superraggedborder}{%
8500   \csuse{@glscmpstyle@superragged}%
8501 }%
```

Backward compatible superraggedheader style.

```
8502 \compatglossarystyle{superraggedheader}{%
8503   \csuse{@glscmpstyle@superragged}%
8504 }%
```

Backward compatible superraggedheaderborder style.

```
8505 \compatglossarystyle{superraggedheaderborder}{%
8506   \csuse{@glscmpstyle@superragged}%
8507 }%
```

Backward compatible superragged3col style.

```
8508 \compatglossarystyle{superragged3col}{%
8509   \renewcommand*{\glossaryentryfield}[5]{%
8510     \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
8511   \renewcommand*{\glossarysubentryfield}[6]{%
8512     &
8513     \glssubentryitem{##2}%
8514     \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8515 }%
```

Backward compatible superragged3colborder style.

```
8516 \compatglossarystyle{superragged3colborder}{%
8517   \csuse{@glscmpstyle@superragged3col}%
8518 }%
```

Backward compatible superragged3colheader style.

```
8519 \compatglossarystyle{superragged3colheader}{%
8520   \csuse{@glscmpstyle@superragged3col}%
8521 }%
```

Backward compatible superragged3colheaderborder style.

```
8522 \compatglossarystyle{superragged3colheaderborder}{%
8523   \csuse{@glscmpstyle@superragged3col}%
8524 }%
```

Backward compatible altsuperragged4col style.

```
8525 \compatglossarystyle{altsuperragged4col}{%
8526   \renewcommand*{\glossaryentryfield}[5]{%
8527     \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8528   \renewcommand*{\glossarysubentryfield}[6]{%
8529     &
8530     \glssubentryitem{##2}%
8531     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8532 }%
```

Backward compatible altsuperragged4colheader style.

```
8533 \compatglossarystyle{altsuperragged4colheader}{%
8534   \csuse{@glscmpstyle@altsuperragged4col}%
8535 }%
```

Backward compatible altsuperragged4colborder style.

```
8536 \compatglossarystyle{altsuperragged4colborder}{%
8537   \csuse{@glscmpstyle@altsuperragged4col}%
8538 }%
```

Backward compatible altsuperragged4colheaderborder style.

```
8539 \compatglossarystyle{altsuperragged4colheaderborder}{%
8540   \csuse{@glscmpstyle@altsuperragged4col}%
8541 }%
```

Backward compatible super style.

```
8542 \compatglossarystyle{super}{%
8543   \renewcommand*{\glossaryentryfield}[5]{%
8544     \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8545   \renewcommand*{\glossarysubentryfield}[6]{%
8546     &
8547     \glssubentryitem{##2}%
8548     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8549 }%
```

Backward compatible superborder style.

```
8550 \compatglossarystyle{superborder}{%
8551   \csuse{@glscmpstyle@super}%
8552 }%
```

Backward compatible superheader style.

```
8553 \compatglossarystyle{superheader}{%
8554   \csuse{@glscmpstyle@super}%
8555 }%
```

Backward compatible superheaderborder style.

```
8556 \compatglossarystyle{superheaderborder}{%
8557   \csuse{@glscmpstyle@super}%
8558 }%
```

Backward compatible super3col style.

```
8559 \compatglossarystyle{super3col}{%
```

```

8560 \renewcommand*{\glossaryentryfield}[5]{%
8561   \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8562 \renewcommand*{\glossarysubentryfield}[6]{%
8563   &
8564   \glssubentryitem{##2}%
8565   \glstarget{##2}{\strut}##4 & ##6\\}%
8566 }%

```

Backward compatible super3colborder style.

```

8567 \compatglossarystyle{super3colborder}{%
8568 \csuse{@glscmpstyle@super3col}%
8569 }%

```

Backward compatible super3colheader style.

```

8570 \compatglossarystyle{super3colheader}{%
8571 \csuse{@glscmpstyle@super3col}%
8572 }%

```

Backward compatible super3colheaderborder style.

```

8573 \compatglossarystyle{super3colheaderborder}{%
8574 \csuse{@glscmpstyle@super3col}%
8575 }%

```

Backward compatible super4col style.

```

8576 \compatglossarystyle{super4col}{%
8577 \renewcommand*{\glossaryentryfield}[5]{%
8578   \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8579 \renewcommand*{\glossarysubentryfield}[6]{%
8580   &
8581   \glssubentryitem{##2}%
8582   \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8583 }%

```

Backward compatible super4colheader style.

```

8584 \compatglossarystyle{super4colheader}{%
8585 \csuse{@glscmpstyle@super4col}%
8586 }%

```

Backward compatible super4colborder style.

```

8587 \compatglossarystyle{super4colborder}{%
8588 \csuse{@glscmpstyle@super4col}%
8589 }%

```

Backward compatible super4colheaderborder style.

```

8590 \compatglossarystyle{super4colheaderborder}{%
8591 \csuse{@glscmpstyle@super4col}%
8592 }%

```

Backward compatible altsuper4col style.

```

8593 \compatglossarystyle{altsuper4col}{%
8594 \csuse{@glscmpstyle@super4col}%
8595 }%

```

Backward compatible altsuper4colheader style.

```
8596 \compatglossarystyle{altsuper4colheader}{%  
8597 \csuse{@glscompstyle@super4col}%  
8598 }%
```

Backward compatible altsuper4colborder style.

```
8599 \compatglossarystyle{altsuper4colborder}{%  
8600 \csuse{@glscompstyle@super4col}%  
8601 }%
```

Backward compatible altsuper4colheaderborder style.

```
8602 \compatglossarystyle{altsuper4colheaderborder}{%  
8603 \csuse{@glscompstyle@super4col}%  
8604 }%
```

6 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.

```
8605 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
8606 \ProvidesPackage{glossaries-accsupp}[2014/01/20 v4.03 (NLCT)  
8607 Experimental glossaries accessibility]
```

Pass all options to glossaries:

```
8608 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
8609 \ProcessOptions
```

`\compatibleglossentry` Override style compatibility macros:

```
8610 \newcommand*{\compatibleglossentry}[2]{%  
8611 \toks@{#2}%  
8612 \protected@edef\@do@glossentry{%  
8613 \noexpand\accsuppglossaryentryfield{#1}%  
8614 {\noexpand\glsnamefont  
8615 {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\endcsname}}%  
8616 {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@desc\endcsname}}%  
8617 {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@symbol\endcsname}}%  
8618 {\the\toks@}%  
8619 }%  
8620 \@do@glossentry  
8621 }
```

`\compatiblesubglossentry`

```
8622 \newcommand*{\compatiblesubglossentry}[3]{%
```



```

8623 \toks@{#3}%
8624 \protected@edef\@do@subglossentry{%
8625   \noexpand\accsuppglossarysubentryfield{\number#1}%
8626   {#2}%
8627   {\noexpand\glsnamefont
8628     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@name\endcsname}}%
8629   {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@desc\endcsname}%
8630   {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@symbol\endcsname}%
8631   {\the\toks@}%
8632 }%
8633 \@do@subglossentry
8634 }

```

Required packages:

```

8635 \RequirePackage{glossaries}
8636 \RequirePackage{accsupp}

```

6.1 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:

```

8637 \define@key{glossentry}{access}{%
8638   \def\@glo@access{#1}%
8639 }

```

textaccess The replacement text corresponding to the text key:

```

8640 \define@key{glossentry}{textaccess}{%
8641   \def\@glo@textaccess{#1}%
8642 }

```

firstaccess The replacement text corresponding to the first key:

```

8643 \define@key{glossentry}{firstaccess}{%
8644   \def\@glo@firstaccess{#1}%
8645 }

```

pluralaccess The replacement text corresponding to the plural key:

```

8646 \define@key{glossentry}{pluralaccess}{%
8647   \def\@glo@pluralaccess{#1}%
8648 }

```

firstpluralaccess The replacement text corresponding to the firstplural key:

```

8649 \define@key{glossentry}{firstpluralaccess}{%
8650   \def\@glo@firstpluralaccess{#1}%
8651 }

```

symbolaccess The replacement text corresponding to the symbol key:

```
8652 \define@key{glossentry}{symbolaccess}{%  
8653   \def\@glo@symbolaccess{#1}%  
8654 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
8655 \define@key{glossentry}{symbolpluralaccess}{%  
8656   \def\@glo@symbolpluralaccess{#1}%  
8657 }
```

descriptionaccess The replacement text corresponding to the description key:

```
8658 \define@key{glossentry}{descriptionaccess}{%  
8659   \def\@glo@descaccess{#1}%  
8660 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
8661 \define@key{glossentry}{descriptionpluralaccess}{%  
8662   \def\@glo@descpluralaccess{#1}%  
8663 }
```

shortaccess The replacement text corresponding to the short key:

```
8664 \define@key{glossentry}{shortaccess}{%  
8665   \def\@glo@shortaccess{#1}%  
8666 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
8667 \define@key{glossentry}{shortpluralaccess}{%  
8668   \def\@glo@shortpluralaccess{#1}%  
8669 }
```

longaccess The replacement text corresponding to the long key:

```
8670 \define@key{glossentry}{longaccess}{%  
8671   \def\@glo@longaccess{#1}%  
8672 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
8673 \define@key{glossentry}{longpluralaccess}{%  
8674   \def\@glo@longpluralaccess{#1}%  
8675 }
```

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={\glsacccsupp{inches}{in}}.

Append these new keys to \@gls@keymap:

```
8676 \appto\@gls@keymap{,%  
8677   {access}{access},%  
8678   {textaccess}{textaccess},%  
8679   {firstaccess}{firstaccess},%
```

```

8680 {pluralaccess}{pluralaccess},%
8681 {firstpluralaccess}{firstpluralaccess},%
8682 {symbolaccess}{symbolaccess},%
8683 {symbolpluralaccess}{symbolpluralaccess},%
8684 {descaccess}{descaccess},%
8685 {descpluralaccess}{descpluralaccess},%
8686 {shortaccess}{shortaccess},%
8687 {shortpluralaccess}{shortpluralaccess},%
8688 {longaccess}{longaccess},%
8689 {longpluralaccess}{longpluralaccess}%
8690 }

```

`\@gls@noaccess` Indicates that no replacement text has been provided.

```

8691 \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):

```

8692 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
8693 \renewcommand*{\@newglossaryentryprehook}{%
8694   \@gls@oldnewglossaryentryprehook
8695   \def\@glo@access{\@glo@symbol}%

```

Initialise the other keys:

```

8696   \def\@glo@textaccess{\@glo@access}%
8697   \def\@glo@firstaccess{\@glo@access}%
8698   \def\@glo@pluralaccess{\@glo@textaccess}%
8699   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
8700   \def\@glo@symbolaccess{\relax}%
8701   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
8702   \def\@glo@descaccess{\relax}%
8703   \def\@glo@descpluralaccess{\@glo@descaccess}%
8704   \def\@glo@shortaccess{\relax}%
8705   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
8706   \def\@glo@longaccess{\relax}%
8707   \def\@glo@longpluralaccess{\@glo@longaccess}%
8708 }

```

Add to the end hook:

```

8709 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
8710 \renewcommand*{\@newglossaryentryposthook}{%
8711   \@gls@oldnewglossaryentryposthook

```

Store the access information:

```

8712   \expandafter
8713     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
8714     \@glo@access}%
8715   \expandafter
8716     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
8717     \@glo@textaccess}%
8718   \expandafter

```

```

8719 \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
8720 \@glo@firstaccess}%
8721 \expandafter
8722 \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
8723 \@glo@pluralaccess}%
8724 \expandafter
8725 \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
8726 \@glo@firstpluralaccess}%
8727 \expandafter
8728 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
8729 \@glo@symbolaccess}%
8730 \expandafter
8731 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
8732 \@glo@symbolpluralaccess}%
8733 \expandafter
8734 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
8735 \@glo@descaccess}%
8736 \expandafter
8737 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
8738 \@glo@descpluralaccess}%
8739 \expandafter
8740 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
8741 \@glo@shortaccess}%
8742 \expandafter
8743 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
8744 \@glo@shortpluralaccess}%
8745 \expandafter
8746 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
8747 \@glo@longaccess}%
8748 \expandafter
8749 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
8750 \@glo@longpluralaccess}%
8751 }

```

6.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

8752 \newcommand*\glsentryaccess[1]{%
8753 \@gls@entry@field{#1}{access}%
8754 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

8755 \newcommand*\glsentrytextaccess[1]{%
8756 \@gls@entry@field{#1}{textaccess}%
8757 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

8758 \newcommand*\glsentryfirstaccess[1]{%
8759 \@gls@entry@field{#1}{firstaccess}%

```

```

8760 }

\glentrypluralaccess  Get the value of the pluralaccess key for the entry with the given label:
8761 \newcommand*{\glentrypluralaccess}[1]{%
8762   \@gls@entry@field{#1}{pluralaccess}%
8763 }

\glentryfirstpluralaccess  Get the value of the firstpluralaccess key for the entry with the given label:
8764 \newcommand*{\glentryfirstpluralaccess}[1]{%
8765   \csname glo@#1@firstpluralaccess\endcsname
8766 }

\glentrysymbolaccess  Get the value of the symbolaccess key for the entry with the given label:
8767 \newcommand*{\glentrysymbolaccess}[1]{%
8768   \@gls@entry@field{#1}{symbolaccess}%
8769 }

\glentrysymbolpluralaccess  Get the value of the symbolpluralaccess key for the entry with the given label:
8770 \newcommand*{\glentrysymbolpluralaccess}[1]{%
8771   \@gls@entry@field{#1}{symbolpluralaccess}%
8772 }

\glentrydescaccess  Get the value of the descriptionaccess key for the entry with the given label:
8773 \newcommand*{\glentrydescaccess}[1]{%
8774   \@gls@entry@field{#1}{descaccess}%
8775 }

\glentrydescpluralaccess  Get the value of the descriptionpluralaccess key for the entry with the given label:
8776 \newcommand*{\glentrydescpluralaccess}[1]{%
8777   \@gls@entry@field{#1}{descaccess}%
8778 }

\glentryshortaccess  Get the value of the shortaccess key for the entry with the given label:
8779 \newcommand*{\glentryshortaccess}[1]{%
8780   \@gls@entry@field{#1}{shortaccess}%
8781 }

\glentryshortpluralaccess  Get the value of the shortpluralaccess key for the entry with the given label:
8782 \newcommand*{\glentryshortpluralaccess}[1]{%
8783   \@gls@entry@field{#1}{shortpluralaccess}%
8784 }

\glentrylongaccess  Get the value of the longaccess key for the entry with the given label:
8785 \newcommand*{\glentrylongaccess}[1]{%
8786   \@gls@entry@field{#1}{longaccess}%
8787 }

```

`trylongpluralaccess` Get the value of the `longpluralaccess` key for the entry with the given label:

```
8788 \newcommand*{\glsentrylongpluralaccess}[1]{%
8789   \@gls@entry@field{#1}{longpluralaccess}%
8790 }
```

`\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.)

```
8791 \newcommand*{\glsaccsupp}[2]{%
8792   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
8793 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
8794 \newcommand*{\xglsaccsupp}[2]{%
8795   \protected@edef\@gls@replacementtext{#1}%
8796   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
8797 }
```

`@gls@access@display`

```
8798 \newcommand*{\@gls@access@display}[2]{%
8799   \protected@edef\@glo@access{#2}%
8800   \ifx\@glo@access\@gls@noaccess
8801     #1%
8802   \else
8803     \xglsaccsupp{\@glo@access}{#1}%
8804   \fi
8805 }
```

`l$nameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
8806 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
8807   \@gls@access@display{#1}{\glsentryaccess{#2}}%
8808 }
```

`ltextaccessdisplay` As above but for the `textaccess` replacement text.

```
8809 \DeclareRobustCommand*{\glsentrytextaccessdisplay}[2]{%
8810   \@gls@access@display{#1}{\glsentrytextaccess{#2}}%
8811 }
```

`pluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```
8812 \DeclareRobustCommand*{\glspluralaccessdisplay}[2]{%
8813   \@gls@access@display{#1}{\glsentrypluralaccess{#2}}%
8814 }
```

`sfirstaccessdisplay` As above but for the `firstaccess` replacement text.

```
8815 \DeclareRobustCommand*{\glsfirstaccessdisplay}[2]{%
8816   \@gls@access@display{#1}{\glsentryfirstaccess{#2}}%
8817 }
```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```

8818 \DeclareRobustCommand*\glfirstpluralaccessdisplay}[2]{%
8819   \@gls@access@display{#1}{\glsentryfirstpluralaccess{#2}}%
8820 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

8821 \DeclareRobustCommand*\glssymbolaccessdisplay}[2]{%
8822   \@gls@access@display{#1}{\glsentrysymbolaccess{#2}}%
8823 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

8824 \DeclareRobustCommand*\glssymbolpluralaccessdisplay}[2]{%
8825   \@gls@access@display{#1}{\glsentrysymbolpluralaccess{#2}}%
8826 }

```

descriptionaccessdisplay As above but for the descriptionaccess replacement text.

```

8827 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
8828   \@gls@access@display{#1}{\glsentrydescaccess{#2}}%
8829 }

```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

8830 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
8831   \@gls@access@display{#1}{\glsentrydescpluralaccess{#2}}%
8832 }

```

shortaccessdisplay As above but for the shortaccess replacement text.

```

8833 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
8834   \@gls@access@display{#1}{\glsentryshortaccess{#2}}%
8835 }

```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```

8836 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
8837   \@gls@access@display{#1}{\glsentryshortpluralaccess{#2}}%
8838 }

```

longaccessdisplay As above but for the longaccess replacement text.

```

8839 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
8840   \@gls@access@display{#1}{\glsentrylongaccess{#2}}%
8841 }

```

pluralaccessdisplay As above but for the longpluralaccess replacement text.

```

8842 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
8843   \@gls@access@display{#1}{\glsentrylongpluralaccess{#2}}%
8844 }

```

\glsaccessdisplay Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

8845 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
8846   \@ifundefined{gls#1accessdisplay}%
8847   {%
8848     \PackageError{glossaries-accsupp}{No accessibility support
8849       for key ‘#1’}{}%
8850   }%
8851   {%
8852     \csname gls#1accessdisplay\endcsname{#2}{#3}%
8853   }%
8854 }

```

ls@default@entryfmt Redefine the default entry format to use accessibility information

```

8855 \renewcommand*\ls@default@entryfmt}[2]{%
8856   \ifdefempty\glscustomtext
8857   {%
8858     \glsifplural
8859     {%
8860       \glscapscase
8861       {%
8862         \ifglsused\glslabel
8863         {%
8864           #2{\glspluralaccessdisplay
8865             {\glsentryplural{\glslabel}}{\glslabel}}%
8866             {\glsdescriptionpluralaccessdisplay
8867               {\glsentrydescplural{\glslabel}}{\glslabel}}%
8868               {\glsymbolpluralaccessdisplay
8869                 {\glsentrysymbolplural{\glslabel}}{\glslabel}}
8870               {\glsinsert}}%
8871             }%
8872             {%
8873               #1{\glsfirstpluralaccessdisplay
8874                 {\glsentryfirstplural{\glslabel}}{\glslabel}}%
8875                 {\glsdescriptionpluralaccessdisplay
8876                   {\glsentrydescplural{\glslabel}}{\glslabel}}%
8877                   {\glsymbolpluralaccessdisplay
8878                     {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
8879                   {\glsinsert}}%
8880                 }%
8881                 {%
8882                 {%
8883                   \ifglsused\glslabel
8884                   {%

```


Subsequent use.

```

8885      #2{\glspluralaccessdisplay
8886          {\Glsentryplural{\glslabel}}{\glslabel}}%
8887          {\glsdescriptionpluralaccessdisplay
8888              {\glsentrydescplural{\glslabel}}{\glslabel}}%
8889          {\glssymbolpluralaccessdisplay
8890              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
8891          {\glsinsert}}%
8892      }%
8893      {%

```

First use

```

8894      #1{\glsfirstpluralaccessdisplay
8895          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
8896          {\glsdescriptionpluralaccessdisplay
8897              {\glsentrydescplural{\glslabel}}{\glslabel}}%
8898          {\glssymbolpluralaccessdisplay
8899              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
8900          {\glsinsert}}%
8901      }%
8902      }%
8903      {%

```

Make all upper case

```

8904      \ifglsused\glslabel
8905      {%

```

Subsequent use

```

8906      \MakeUppercase{%
8907          #2{\glspluralaccessdisplay
8908              {\glsentryplural{\glslabel}}{\glslabel}}%
8909              {\glsdescriptionpluralaccessdisplay
8910                  {\glsentrydescplural{\glslabel}}{\glslabel}}%
8911                  {\glssymbolpluralaccessdisplay
8912                      {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
8913                      {\glsinsert}}}%
8914      }%
8915      {%

```

First use

```

8916      \MakeUppercase{%
8917          #1{\glsfirstpluralaccessdisplay
8918              {\glsentryfirstplural{\glslabel}}{\glslabel}}%
8919              {\glsdescriptionpluralaccessdisplay
8920                  {\glsentrydescplural{\glslabel}}{\glslabel}}%
8921                  {\glssymbolpluralaccessdisplay
8922                      {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
8923                      {\glsinsert}}}%
8924      }%
8925      }%
8926      }%

```

```

8927      {%
      Singular form
8928      \glscapscase
8929      {%
      Don't adjust case
8930      \ifglused\glslabel
8931      {%
      Subsequent use
8932      #2{\glstextaccessdisplay
8933          {\glentrytext{\glslabel}}{\glslabel}}%
8934          {\gldescriptionaccessdisplay
8935              {\glentrydesc{\glslabel}}{\glslabel}}%
8936              {\glssymbolaccessdisplay
8937                  {\glentrysymbol{\glslabel}}{\glslabel}}%
8938                  {\glinsert}%
8939      }%
8940      {%
      First use
8941      #1{\glfirstaccessdisplay
8942          {\glentryfirst{\glslabel}}{\glslabel}}%
8943          {\gldescriptionaccessdisplay
8944              {\glentrydesc{\glslabel}}{\glslabel}}%
8945              {\glssymbolaccessdisplay
8946                  {\glentrysymbol{\glslabel}}{\glslabel}}%
8947                  {\glinsert}%
8948      }%
8949      }%
8950      {%
      Make first letter upper case
8951      \ifglused\glslabel
8952      {%
      Subsequent use
8953      #2{\Glstextaccessdisplay
8954          {\Glentrytext{\glslabel}}{\glslabel}}%
8955          {\Glsdescriptionaccessdisplay
8956              {\Glentrydesc{\glslabel}}{\glslabel}}%
8957              {\Glsymbolaccessdisplay
8958                  {\Glentrysymbol{\glslabel}}{\glslabel}}%
8959                  {\Glsinsert}%
8960      }%
8961      {%
      First use
8962      #1{\Glsfirstaccessdisplay
8963          {\Glentryfirst{\glslabel}}{\glslabel}}%
8964          {\Glsdescriptionaccessdisplay

```

```

8965         {\glentrydesc{\glslabel}}{\glslabel}}%
8966     {\glssymbolaccessdisplay
8967         {\glentrysymbol{\glslabel}}{\glslabel}}%
8968     {\glsinsert}%
8969 }%
8970 }%
8971 {%

    Make all upper case
8972     \ifglsused\glslabel
8973     {%

        Subsequent use
8974         \MakeUppercase{%
8975             #2{\glstextaccessdisplay
8976                 {\glentrytext{\glslabel}}{\glslabel}}%
8977                 {\glsdescriptionaccessdisplay
8978                     {\glentrydesc{\glslabel}}{\glslabel}}%
8979                 {\glssymbolaccessdisplay
8980                     {\glentrysymbol{\glslabel}}{\glslabel}}%
8981                 {\glsinsert}}%
8982             }%
8983         {%

            First use
8984             \MakeUppercase{%
8985                 #1{\glsfirstaccessdisplay
8986                     {\glentryfirst{\glslabel}}{\glslabel}}%
8987                     {\glsdescriptionaccessdisplay
8988                         {\glentrydesc{\glslabel}}{\glslabel}}%
8989                     {\glssymbolaccessdisplay
8990                         {\glentrysymbol{\glslabel}}{\glslabel}}%
8991                     {\glsinsert}}%
8992                 }%
8993             }%
8994         }%
8995     }%
8996     {%

        Custom text provided in \glsdisp
8997         \ifglsused{\glslabel}%
8998         {%

            Subsequent use
8999             #2{\glscustomtext}%
9000             {\glsdescriptionaccessdisplay
9001                 {\glentrydesc{\glslabel}}{\glslabel}}%
9002             {\glssymbolaccessdisplay
9003                 {\glentrysymbol{\glslabel}}{\glslabel}}%
9004             {\glsinsert}%
9005         }%
9006     {%

```

First use

```

9007      #1{\glscustomtext}%
9008      {\glsdescriptionaccessdisplay
9009       {\glsentrydesc{\glslabel}}{\glslabel}}%
9010      {\glssymbolaccessdisplay
9011       {\glsentrysymbol{\glslabel}}{\glslabel}}%
9012      {\glsinsert}%
9013    }%
9014  }%
9015}

```

`\glsgenentryfmt` Redefine to use accessibility information.

```

9016 \renewcommand*{\glsgenentryfmt}{%
9017   \ifdefempty\glscustomtext
9018     {%
9019       \glsifplural
9020     {%

```

Plural form

```

9021       \glscapscase
9022     {%

```

Don't adjust case

```

9023       \ifglsused\glslabel
9024     {%

```

Subsequent use

```

9025       \glspluralaccessdisplay
9026       {\glsentryplural{\glslabel}}{\glslabel}%
9027       \glsinsert
9028     }%
9029     {%

```

First use

```

9030       \glsfirstpluralaccessdisplay
9031       {\glsentryfirstplural{\glslabel}}{\glslabel}%
9032       \glsinsert
9033     }%
9034     }%
9035     {%

```

Make first letter upper case

```

9036       \ifglsused\glslabel
9037     {%

```

Subsequent use.

```

9038       \glspluralaccessdisplay
9039       {\Glsentryplural{\glslabel}}{\glslabel}%
9040       \glsinsert
9041     }%
9042     {%

```

First use

```
9043      \glsfirstpluralaccessdisplay
9044      {\Glsentryfirstplural{\glslabel}}{\glslabel}%
9045      \glsinsert
9046      }%
9047      }%
9048      {%
```

Make all upper case

```
9049      \ifglsused\glslabel
9050      {%
```

Subsequent use

```
9051      \glspluralaccessdisplay
9052      {\mfirstucMakeUppercase{\glsentryplural{\glslabel}}}%
9053      {\glslabel}%
9054      \mfirstucMakeUppercase{\glsinsert}%
9055      }%
9056      {%
```

First use

```
9057      \glsfirstpluralaccessdisplay
9058      {\mfirstucMakeUppercase{\glsentryfirstplural{\glslabel}}}%
9059      {\glslabel}%
9060      \mfirstucMakeUppercase{\glsinsert}%
9061      }%
9062      }%
9063      }%
9064      {%
```

Singular form

```
9065      \glscapscase
9066      {%
```

Don't adjust case

```
9067      \ifglsused\glslabel
9068      {%
```

Subsequent use

```
9069      \glstextaccessdisplay{\glsentrytext{\glslabel}}{\glslabel}%
9070      \glsinsert
9071      }%
9072      {%
```

First use

```
9073      \glsfirstaccessdisplay{\glsentryfirst{\glslabel}}{\glslabel}%
9074      \glsinsert
9075      }%
9076      }%
9077      {%
```

Make first letter upper case

```
9078      \ifglused\glslabel
9079      {%
```

Subsequent use

```
9080      \glstextaccessdisplay{\Glsentrytext{\glslabel}}{\glslabel}%
9081      \glsinsert
9082      }%
9083      {%
```

First use

```
9084      \glsfirstaccessdisplay{\Glsentryfirst{\glslabel}}{\glslabel}%
9085      \glsinsert
9086      }%
9087      }%
9088      {%
```

Make all upper case

```
9089      \ifglused\glslabel
9090      {%
```

Subsequent use

```
9091      \glstextaccessdisplay
9092      {\mfirstucMakeUppercase{\glsentrytext{\glslabel}}}{\glslabel}%
9093      \mfirstucMakeUppercase{\glsinsert}%
9094      }%
9095      {%
```

First use

```
9096      \glsfirstaccessdisplay
9097      {\mfirstucMakeUppercase{\glsentryfirst{\glslabel}}}{\glslabel}%
9098      \mfirstucMakeUppercase{\glsinsert}%
9099      }%
9100      }%
9101      }%
9102      }%
9103      {%
```

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.

```
9104      \glscustomtext\glsinsert
9105      }%
9106 }
```

`\glsacft` Redefine to include accessibility information.

```
9107 \renewcommand*{\glsacft}{%
9108   \ifdefempty\glscustomtext
9109   {%
9110     \ifglused\glslabel
9111     {%
```

Subsequent use:

```
9112      \glsifplural
9113      {%
```

Subsequent plural form:

```
9114      \glscapscase
9115      {%
```

Subsequent plural form, don't adjust case:

```
9116      \acronymfont
9117      {\glsshortpluralaccessdisplay
9118      {\glentryshortpl{\glslabel}}{\glslabel}}%
9119      \glsinsert
9120      }%
9121      {%
```

Subsequent plural form, make first letter upper case:

```
9122      \acronymfont
9123      {\glsshortpluralaccessdisplay
9124      {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9125      \glsinsert
9126      }%
9127      {%
```

Subsequent plural form, all caps:

```
9128      \mfirstucMakeUppercase
9129      {\acronymfont
9130      {\glsshortpluralaccessdisplay
9131      {\glentryshortpl{\glslabel}}{\glslabel}}%
9132      \glsinsert}%
9133      }%
9134      }%
9135      {%
```

Subsequent singular form

```
9136      \glscapscase
9137      {%
```

Subsequent singular form, don't adjust case:

```
9138      \acronymfont
9139      {\glsshortaccessdisplay{\glentryshort{\glslabel}}{\glslabel}}%
9140      \glsinsert
9141      }%
9142      {%
```

Subsequent singular form, make first letter upper case:

```
9143      \acronymfont
9144      {\glsshortaccessdisplay{\Glsentryshort{\glslabel}}{\glslabel}}%
9145      \glsinsert
9146      }%
9147      {%
```

Subsequent singular form, all caps:

```

9148         \mfirstucMakeUppercase
9149         {\acronymfont{%
9150         \glsshortaccessdisplay{\glstryshort{\glslabel}}{\glslabel}}}%
9151         \glsinsert}%
9152     }%
9153 }%
9154 }%
9155 {%

```

First use:

```

9156     \glsifplural
9157     {%

```

First use plural form:

```

9158     \glscapscase
9159     {%

```

First use plural form, don't adjust case:

```

9160     \genplacrfullformat{\glslabel}{\glsinsert}%
9161     }%
9162     {%

```

First use plural form, make first letter upper case:

```

9163     \Genplacrfullformat{\glslabel}{\glsinsert}%
9164     }%
9165     {%

```

First use plural form, all caps:

```

9166     \mfirstucMakeUppercase
9167     {\genplacrfullformat{\glslabel}{\glsinsert}}}%
9168     }%
9169     }%
9170     {%

```

First use singular form

```

9171     \glscapscase
9172     {%

```

First use singular form, don't adjust case:

```

9173     \genacrfullformat{\glslabel}{\glsinsert}%
9174     }%
9175     {%

```

First use singular form, make first letter upper case:

```

9176     \Genacrfullformat{\glslabel}{\glsinsert}%
9177     }%
9178     {%

```

First use singular form, all caps:

```

9179     \mfirstucMakeUppercase
9180     {\genacrfullformat{\glslabel}{\glsinsert}}}%
9181     }%

```



```

9182     }%
9183     }%
9184     }%
9185     {%

```

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.

```

9186     \glscustomtext
9187     }%
9188 }

```

`\genacrfullformat` Redefine to include accessibility information.

```

9189 \renewcommand*{\genacrfullformat}[2]{%
9190     \glslongaccessdisplay{\glentrylong{#1}}{#1}#2\space
9191     (\glsshortaccessdisplay{\protect\firstacronymfont{\glentryshort{#1}}}{#1})%
9192 }

```

`\Genacrfullformat` Redefine to include accessibility information.

```

9193 \renewcommand*{\Genacrfullformat}[2]{%
9194     \glslongaccessdisplay{\Glentrylong{#1}}{#1}#2\space
9195     (\glsshortaccessdisplay{\protect\firstacronymfont{\Glentryshort{#1}}}{#1})%
9196 }

```

`\genplacrfullformat` Redefine to include accessibility information.

```

9197 \renewcommand*{\genplacrfullformat}[2]{%
9198     \glslongpluralaccessdisplay{\glentrylongpl{#1}}{#1}#2\space
9199     (\glsshortpluralaccessdisplay
9200         {\protect\firstacronymfont{\glentryshortpl{#1}}}{#1})%
9201 }

```

`\Genplacrfullformat` Redefine to include accessibility information.

```

9202 \renewcommand*{\Genplacrfullformat}[2]{%
9203     \glslongpluralaccessdisplay{\Glentrylongpl{#1}}{#1}#2\space
9204     (\glsshortpluralaccessdisplay
9205         {\protect\firstacronymfont{\Glentryshortpl{#1}}}{#1})%
9206 }

```

`\@acrshort`

```

9207 \def\@acrshort#1#2[#3]{%
9208     \glsoifexists{#2}%
9209     {%
9210         \edef\@glo@type{\glentrytype{#2}}%
9211         \let\glsifplural\@secondoftwo
9212         \let\glscapscase\@firstofthree
9213         \let\glsinsert\@empty
9214         \def\glscustomtext{%
9215             \acronymfont{\glsshortaccessdisplay{\glentryshort{#2}}{#2}}#3%
9216         }%

```

```

Call \@gls@link
9217   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9218   }%
9219 }

```

\@Acrshort

```

9220 \def\@Acrshort#1#2[#3]{%
9221   \glsdoifexists{#2}%
9222   {%
9223     \edef\@glo@type{\glentrytype{#2}}%

9224     \let\glsifplural\@secondoftwo
9225     \let\glscapscase\@secondofthree
9226     \let\glsinsert\@empty
9227     \def\glscustomtext{%
9228       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
9229     }%

```

```

Call \@gls@link
9230   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9231   }%
9232 }

```

\@ACRshort

```

9233 \def\@ACRshort#1#2[#3]{%
9234   \glsdoifexists{#2}%
9235   {%
9236     \edef\@glo@type{\glentrytype{#2}}%

9237     \let\glsifplural\@secondoftwo
9238     \let\glscapscase\@thirdofthree
9239     \let\glsinsert\@empty
9240     \def\glscustomtext{%
9241       \acronymfont{\glsshortaccessdisplay
9242         {\MakeUppercase{\glentryshort{#2}}}{#2}}#3%
9243     }%

```

```

Call \@gls@link
9244   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9245   }%
9246 }

```

\@acrlong

```

9247 \def\@acrlong#1#2[#3]{%
9248   \glsdoifexists{#2}%
9249   {%
9250     \edef\@glo@type{\glentrytype{#2}}%

```

```

9251 \let\glsifplural\@secondoftwo
9252 \let\glscapscase\@firstofthree
9253 \let\glsinsert\@empty
9254 \def\glscustomtext{%
9255 \acronymfont{\glslongaccessdisplay{\glsentrylong{#2}}{#2}}#3%
9256 }%

```

Call \@gls@link

```

9257 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9258 }%
9259 }

```

\@Acrlong

```

9260 \def\@Acrlong#1#2[#3]{%
9261 \glsdoifexists{#2}%
9262 {%
9263 \edef\@glo@type{\glsentrytype{#2}}%

9264 \let\glsifplural\@secondoftwo
9265 \let\glscapscase\@firstofthree
9266 \let\glsinsert\@empty
9267 \def\glscustomtext{%
9268 \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
9269 }%

```

Call \@gls@link

```

9270 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9271 }%
9272 }

```

\@ACRlong

```

9273 \def\@ACRlong#1#2[#3]{%
9274 \glsdoifexists{#2}%
9275 {%
9276 \edef\@glo@type{\glsentrytype{#2}}%

9277 \let\glsifplural\@secondoftwo
9278 \let\glscapscase\@firstofthree
9279 \let\glsinsert\@empty
9280 \def\glscustomtext{%
9281 \acronymfont{\glslongaccessdisplay{%
9282 \MakeUppercase{\glsentrylong{#2}}}{#2}}#3}%
9283 }%

```

Call \@gls@link

```

9284 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9285 }%
9286 }

```

6.3 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.

```
9287 \renewcommand*{\glossentryname}[1]{%
9288   \glsdoifexists{#1}%
9289   {%
9290     \glsnamefont{\glsnameaccessdisplay{\glsentryname{#1}}{#1}}%
9291   }%
9292 }

9293 \renewcommand*{\glossentryname}[1]{%
9294   \glsdoifexists{#1}%
9295   {%
9296     \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
9297   }%
9298 }

9299 \renewcommand*{\glossentrydesc}[1]{%
9300   \glsdoifexists{#1}%
9301   {%
9302     \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
9303   }%
9304 }

9305 \renewcommand*{\Glossentrydesc}[1]{%
9306   \glsdoifexists{#1}%
9307   {%
9308     \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
9309   }%
9310 }

9311 \renewcommand*{\glossentrysymbol}[1]{%
9312   \glsdoifexists{#1}%
9313   {%
9314     \glssymbolaccessdisplay{\glsentrysymbol{#1}}{#1}%
9315   }%
9316 }

9317 \renewcommand*{\Glossentrysymbol}[1]{%
9318   \glsdoifexists{#1}%
9319   {%
9320     \glssymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
9321   }%
9322 }
```

pglossaryentryfield

```

9323 \newcommand*{\accsuppglossaryentryfield}[5]{%
9324   \glossaryentryfield{#1}%
9325   {\glnameaccessdisplay{#2}{#1}}%
9326   {\gldescriptionaccessdisplay{#3}{#1}}%
9327   {\glssymbolaccessdisplay{#4}{#1}}{#5}%
9328 }
```

ossarysubentryfield

```

9329 \newcommand*{\accsuppglossarysubentryfield}[6]{%
9330   \glossarysubentryfield{#1}{#2}%
9331   {\glnameaccessdisplay{#3}{#2}}%
9332   {\gldescriptionaccessdisplay{#4}{#2}}%
9333   {\glssymbolaccessdisplay{#5}{#2}}{#6}%
9334 }
```

6.4 Acronyms

Redefine acronym styles provided by glossaries:

long-short *<long>* (*<short>*) acronym style.

```

9335 \renewacronymstyle{long-short}%
9336 {%
```

Check for long form in case this is a mixed glossary.

```

9337   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
9338 }%
9339 {%
9340   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
9341   \renewcommand*{\genacrfullformat}[2]{%
9342     \glslongaccessdisplay{\glentrylong{##1}}{##1}##2\space
9343     (\glsshortaccessdisplay
9344       {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
9345   }%
9346   \renewcommand*{\Genacrfullformat}[2]{%
9347     \glslongaccessdisplay{\Glentrylong{##1}}{##1}##2\space
9348     (\glsshortaccessdisplay
9349       {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
9350   }%
9351   \renewcommand*{\genplacrfullformat}[2]{%
9352     \glslongpluralaccessdisplay{\glentrylongpl{##1}}{##1}##2\space
9353     (\glsshortpluralaccessdisplay
9354       {\protect\firstacronymfont{\glentryshortpl{##1}}}{##1})%
9355   }%
9356   \renewcommand*{\Genplacrfullformat}[2]{%
9357     \glslongpluralaccessdisplay{\Glentrylongpl{##1}}{##1}##2\space
9358     (\glsshortpluralaccessdisplay
9359       {\protect\firstacronymfont{\glentryshortpl{##1}}}{##1})%
9360   }%
```

```

9361 \renewcommand*{\acronymentry}[1]{%
9362     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
9363 \renewcommand*{\acronymsort}[2]{##1}%
9364 \renewcommand*{\acronymfont}[1]{##1}%
9365 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
9366 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
9367 }

```

short-long *<short>* (*<long>*) acronym style.

```

9368 \renewacronymstyle{short-long}%
9369 {%
    Check for long form in case this is a mixed glossary.
9370     \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
9371 }%
9372 {%
9373     \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
9374     \renewcommand*{\genacrfullformat}[2]{%
9375         \glsshortaccessdisplay
9376         {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2\space
9377         (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
9378     }%
9379     \renewcommand*{\Genacrfullformat}[2]{%
9380         \glsshortaccessdisplay
9381         {\protect\firstacronymfont{\Glsentryshort{##1}}}{##1}##2\space
9382         (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
9383     }%
9384     \renewcommand*{\genplacrfullformat}[2]{%
9385         \glsshortpluralaccessdisplay
9386         {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2\space
9387         (\glslongpluralaccessdisplay
9388         {\glsentrylongpl{##1}}{##1})%
9389     }%
9390     \renewcommand*{\Genplacrfullformat}[2]{%
9391         \glsshortpluralaccessdisplay
9392         {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2\space
9393         (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})%
9394     }%
9395     \renewcommand*{\acronymentry}[1]{%
9396         \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%
9397     \renewcommand*{\acronymsort}[2]{##1}%
9398     \renewcommand*{\acronymfont}[1]{##1}%
9399     \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
9400     \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
9401 }

```

long-short-desc *<long>* (*<short>*) acronym style that has an accompanying description (which the user needs to supply).

```

9402 \renewacronymstyle{long-short-desc}%

```

```

9403 {%
9404   \GlsUseAcrEntryDispStyle{long-short}%
9405 }%
9406 {%
9407   \GlsUseAcrStyleDefs{long-short}%
9408   \renewcommand*{\GenericAcronymFields}{}%
9409   \renewcommand*{\acronymsort}[2]{##2}%
9410   \renewcommand*{\acronymentry}[1]{%
9411     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9412     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
9413 }

```

long-sc-short-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

9414 \renewacronymstyle{long-sc-short-desc}%
9415 {%
9416   \GlsUseAcrEntryDispStyle{long-sc-short}%
9417 }%
9418 {%
9419   \GlsUseAcrStyleDefs{long-sc-short}%
9420   \renewcommand*{\GenericAcronymFields}{}%
9421   \renewcommand*{\acronymsort}[2]{##2}%
9422   \renewcommand*{\acronymentry}[1]{%
9423     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9424     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
9425 }

```

long-sm-short-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

9426 \renewacronymstyle{long-sm-short-desc}%
9427 {%
9428   \GlsUseAcrEntryDispStyle{long-sm-short}%
9429 }%
9430 {%
9431   \GlsUseAcrStyleDefs{long-sm-short}%
9432   \renewcommand*{\GenericAcronymFields}{}%
9433   \renewcommand*{\acronymsort}[2]{##2}%
9434   \renewcommand*{\acronymentry}[1]{%
9435     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9436     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
9437 }

```

short-long-desc *<short>* ({<long>}) acronym style that has an accompanying description (which the user needs to supply).

```

9438 \renewacronymstyle{short-long-desc}%
9439 {%
9440   \GlsUseAcrEntryDispStyle{short-long}%
9441 }%
9442 {%

```

```

9443 \GlsUseAcrStyleDefs{short-long}%
9444 \renewcommand*{\GenericAcronymFields}{}%
9445 \renewcommand*{\acronymsort}[2]{##2}%
9446 \renewcommand*{\acronymentry}[1]{%
9447     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9448     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
9449 }

```

sc-short-long-desc *⟨long⟩* (\textsc{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

9450 \renewacronymstyle{sc-short-long-desc}%
9451 {%
9452     \GlsUseAcrEntryDispStyle{sc-short-long}%
9453 }%
9454 {%
9455     \GlsUseAcrStyleDefs{sc-short-long}%
9456     \renewcommand*{\GenericAcronymFields}{}%
9457     \renewcommand*{\acronymsort}[2]{##2}%
9458     \renewcommand*{\acronymentry}[1]{%
9459         \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9460         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
9461 }

```

sm-short-long-desc *⟨long⟩* (\textsmaller{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

9462 \renewacronymstyle{sm-short-long-desc}%
9463 {%
9464     \GlsUseAcrEntryDispStyle{sm-short-long}%
9465 }%
9466 {%
9467     \GlsUseAcrStyleDefs{sm-short-long}%
9468     \renewcommand*{\GenericAcronymFields}{}%
9469     \renewcommand*{\acronymsort}[2]{##2}%
9470     \renewcommand*{\acronymentry}[1]{%
9471         \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9472         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
9473 }

```

dua *⟨long⟩* only acronym style.

```

9474 \renewacronymstyle{dua}%
9475 {%

```

Check for long form in case this is a mixed glossary.

```

9476 \ifdefempty\glscustomtext
9477 {%
9478     \ifglshaslong{\glslabel}%
9479     {%
9480         \glsifplural
9481         {%

```


Plural form:

```
9482      \glscapscase
9483      {%
```

Plural form, don't adjust case:

```
9484      \glslongpluralaccessdisplay{\glsentrylongpl{\glslabel}}{\glslabel}%
9485      \glsinsert
9486      }%
9487      {%
```

Plural form, make first letter upper case:

```
9488      \glslongpluralaccessdisplay{\Glsentrylongpl{\glslabel}}{\glslabel}%
9489      \glsinsert
9490      }%
9491      {%
```

Plural form, all caps:

```
9492      \glslongpluralaccessdisplay
9493      {\mfirstucMakeUppercase{\glsentrylongpl{\glslabel}}}{\glslabel}%
9494      \mfirstucMakeUppercase{\glsinsert}%
9495      }%
9496      }%
9497      {%
```

Singular form

```
9498      \glscapscase
9499      {%
```

Singular form, don't adjust case:

```
9500      \glslongaccessdisplay{\glsentrylong{\glslabel}}{\glslabel}\glsinsert
9501      }%
9502      {%
```

Subsequent singular form, make first letter upper case:

```
9503      \glslongaccessdisplay{\Glsentrylong{\glslabel}}{\glslabel}\glsinsert
9504      }%
9505      {%
```

Subsequent singular form, all caps:

```
9506      \glslongaccessdisplay
9507      {\mfirstucMakeUppercase
9508      {\glsentrylong{\glslabel}\glsinsert}}{\glslabel}%
9509      \mfirstucMakeUppercase{\glsinsert}%
9510      }%
9511      }%
9512      }%
9513      {%
```

Not an acronym:

```
9514      \glsgenentryfmt
9515      }%
9516      }%
```

```

9517 {\glscustomtext\glsinsert}%
9518 }%
9519 {%
9520 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
9521 \renewcommand*{\acrfullfmt}[3]{%
9522   \glslink[##1]{##2}{%
9523     \glslongaccessdisplay{\glsentrylong{##2}}{##2}##3\space
9524     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2})}%
9525 \renewcommand*{\Acrfullfmt}[3]{%
9526   \glslink[##1]{##2}{%
9527     \glslongaccessdisplay{\Glsentrylong{##2}}{##2}##3\space
9528     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2})}%
9529 \renewcommand*{\ACRfullfmt}[3]{%
9530   \glslink[##1]{##2}{%
9531     \glslongaccessdisplay
9532       {\mfirstucMakeUppercase{\glsentrylong{##2}}{##2}##3\space
9533       (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2})}}}%
9534 \renewcommand*{\acrfullplfmt}[3]{%
9535   \glslink[##1]{##2}{%
9536     \glslongpluralaccessdisplay
9537       {\glsentrylongpl{##2}}{##2}##3\space
9538     (\glsshortpluralaccessdisplay
9539       {\acronymfont{\glsentryshortpl{##2}}}{##2})}%
9540 \renewcommand*{\Acrfullplfmt}[3]{%
9541   \glslink[##1]{##2}{%
9542     \glslongpluralaccessdisplay
9543       {\Glsentrylongpl{##2}}{##2}##3\space
9544     (\glsshortpluralaccessdisplay
9545       {\acronymfont{\glsentryshortpl{##2}}}{##2})}%
9546 \renewcommand*{\ACRfullplfmt}[3]{%
9547   \glslink[##1]{##2}{%
9548     \glslongpluralaccessdisplay
9549       {\mfirstucMakeUppercase{\glsentrylongpl{##2}}{##2}##3\space
9550     (\glsshortpluralaccessdisplay
9551       {\acronymfont{\glsentryshortpl{##2}}}{##2})}}}%
9552 \renewcommand*{\glsentryfull}[1]{%
9553   \glslongaccessdisplay{\glsentrylong{##1}}\space
9554   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})%
9555 }%
9556 \renewcommand*{\Glsentryfull}[1]{%
9557   \glslongaccessdisplay{\Glsentrylong{##1}}{##1}\space
9558   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})%
9559 }%
9560 \renewcommand*{\glsentryfullpl}[1]{%
9561   \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}\space
9562   (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}}{##1})%
9563 }%
9564 \renewcommand*{\Glsentryfullpl}[1]{%
9565   \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}\space

```

```

9566      (\glsshortpluralaccessdisplay{\acronymfont{\glsenentryshortpl{##1}}}{##1}})%
9567 }%
9568 \renewcommand*{\acronymentry}[1]{%
9569     \glsshortaccessdisplay{\acronymfont{\glsenentryshort{##1}}}{##1}}%
9570 \renewcommand*{\acronymsort}[2]{##1}%
9571 \renewcommand*{\acronymfont}[1]{##1}%
9572 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
9573 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

9574 \renewacronymstyle{dua-desc}%
9575 {%
9576     \GlsUseAcrEntryDispStyle{dua}%
9577 }%
9578 {%
9579     \GlsUseAcrStyleDefs{dua}%
9580 \renewcommand*{\GenericAcronymFields}{}%
9581 \renewcommand*{\acronymentry}[1]{%
9582     \glslongaccessdisplay{\acronymfont{\glsenentrylong{##1}}}{##1}}%
9583 \renewcommand*{\acronymsort}[2]{##2}%
9584 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

9585 \renewacronymstyle{footnote}%
9586 {%

```

Check for long form in case this is a mixed glossary.

```

9587 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
9588 }%
9589 {%
9590 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

9591 \glshyperfirstfalse
9592 \renewcommand*{\genacrfullformat}[2]{%
9593     \glsshortaccessdisplay
9594     {\protect\firstacronymfont{\glsenentryshort{##1}}}{##1}##2%
9595     \protect\footnote{\glslongaccessdisplay{\glsenentrylong{##1}}}{##1}}%
9596 }%
9597 \renewcommand*{\Genacrfullformat}[2]{%
9598     \glsshortaccessdisplay
9599     {\firstacronymfont{\glsenentryshort{##1}}}{##1}##2%
9600     \protect\footnote{\glslongaccessdisplay{\glsenentrylong{##1}}}{##1}}%
9601 }%
9602 \renewcommand*{\genplacrfullformat}[2]{%
9603     \glsshortpluralaccessdisplay
9604     {\protect\firstacronymfont{\glsenentryshortpl{##1}}}{##1}##2%
9605     \protect\footnote{\glslongpluralaccessdisplay{\glsenentrylongpl{##1}}}{##1}}%
9606 }%
9607 \renewcommand*{\Genplacrfullformat}[2]{%

```

```

9608 \glsshortpluralaccessdisplay
9609 {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2%
9610 \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}}%
9611 }%
9612 \renewcommand*{\acronymentry}[1]{%
9613 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%
9614 \renewcommand*{\acronymsort}[2]{##1}%
9615 \renewcommand*{\acronymfont}[1]{##1}%
9616 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%

```

Don't use footnotes for \acrfull:

```

9617 \renewcommand*{\acrfullfmt}[3]{%
9618 \glslink[##1]{##2}{%
9619 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}##3\space
9620 (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}}%
9621 \renewcommand*{\Acrfullfmt}[3]{%
9622 \glslink[##1]{##2}{%
9623 \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##2}}}{##2}##3\space
9624 (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}}%
9625 \renewcommand*{\ACRfullfmt}[3]{%
9626 \glslink[##1]{##2}{%
9627 \glsshortaccessdisplay
9628 {\mfirstucMakeUppercase
9629 {\acronymfont{\glsentryshort{##2}}}{##2}##3\space
9630 (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}}}%
9631 \renewcommand*{\acrfullplfmt}[3]{%
9632 \glslink[##1]{##2}{%
9633 \glsshortpluralaccessdisplay
9634 {\acronymfont{\glsentryshortpl{##2}}}{##2}##3\space
9635 (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}}}%
9636 \renewcommand*{\Acrfullplfmt}[3]{%
9637 \glslink[##1]{##2}{%
9638 \glsshortpluralaccessdisplay
9639 {\acronymfont{\Glsentryshortpl{##2}}}{##2}##3\space
9640 (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}}}%
9641 \renewcommand*{\ACRfullplfmt}[3]{%
9642 \glslink[##1]{##2}{%
9643 \glsshortpluralaccessdisplay
9644 {\mfirstucMakeUppercase
9645 {\acronymfont{\glsentryshortpl{##2}}}{##2}##3\space
9646 (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}}}%

```

Similarly for \glsentryfull etc:

```

9647 \renewcommand*{\glsentryfull}[1]{%
9648 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}\space
9649 (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
9650 \renewcommand*{\Glsentryfull}[1]{%
9651 \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##1}}}{##1}\space
9652 (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
9653 \renewcommand*{\glsentryfullpl}[1]{%

```

```

9654 \glsshortpluralaccessdisplay
9655 {\acronymfont{\glentryshortpl{##1}}{##1}\space
9656 (\glslongpluralaccessdisplay{\glentrylongpl{##1}}{##1})}%
9657 \renewcommand*\Glsentryfullpl}[1]{%
9658 \glsshortpluralaccessdisplay
9659 {\acronymfont{\Glsentryshortpl{##1}}{##1}\space
9660 (\glslongpluralaccessdisplay{\glentrylongpl{##1}}{##1})}%
9661 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

9662 \renewacronymstyle{footnote-sc}%
9663 {%
9664 \GlsUseAcrEntryDisplayStyle{footnote}%
9665 }%
9666 {%
9667 \GlsUseAcrStyleDefs{footnote}%
9668 \renewcommand{\acronymentry}[1]{%
9669 \glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}{##1}}
9670 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
9671 \renewcommand*\acrpluralsuffix{\glstextup{\glspluralsuffix}}%
9672 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

9673 \renewacronymstyle{footnote-sm}%
9674 {%
9675 \GlsUseAcrEntryDisplayStyle{footnote}%
9676 }%
9677 {%
9678 \GlsUseAcrStyleDefs{footnote}%
9679 \renewcommand{\acronymentry}[1]{%
9680 \glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}{##1}}
9681 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
9682 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
9683 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

9684 \renewacronymstyle{footnote-desc}%
9685 {%
9686 \GlsUseAcrEntryDisplayStyle{footnote}%
9687 }%
9688 {%
9689 \GlsUseAcrStyleDefs{footnote}%
9690 \renewcommand*\GenericAcronymFields{}}%
9691 \renewcommand*\acronymsort}[2]{##2}%
9692 \renewcommand*\acronymentry}[1]{%
9693 \glslongaccessdisplay{\glentrylong{##1}}{##1}\space
9694 (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}{##1}})%
9695 }

```

footnote-sc-desc \textsc{\<short>}\footnote{\<long>} acronym style that has an accompanying description (which the user needs to supply).

```

9696 \renewacronymstyle{footnote-sc-desc}%
9697 {%
9698   \GlsUseAcrEntryDispStyle{footnote-sc}%
9699 }%
9700 {%
9701   \GlsUseAcrStyleDefs{footnote-sc}%
9702   \renewcommand*\<GenericAcronymFields>{}%
9703   \renewcommand*\<acronymsort>[2]{##2}%
9704   \renewcommand*\<acronymentry>[1]{%
9705     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9706     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1)}}%
9707 }
```

footnote-sm-desc \textsmaller{\<short>}\footnote{\<long>} acronym style that has an accompanying description (which the user needs to supply).

```

9708 \renewacronymstyle{footnote-sm-desc}%
9709 {%
9710   \GlsUseAcrEntryDispStyle{footnote-sm}%
9711 }%
9712 {%
9713   \GlsUseAcrStyleDefs{footnote-sm}%
9714   \renewcommand*\<GenericAcronymFields>{}%
9715   \renewcommand*\<acronymsort>[2]{##2}%
9716   \renewcommand*\<acronymentry>[1]{%
9717     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
9718     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1)}}%
9719 }
```

Use \newacronymhook to modify the key list to set the access text to the long version by default.

```

9720 \renewcommand*\<newacronymhook>{%
9721   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
9722     \the\glskeylisttok}%
9723   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
9724 }
```

defaultNewAcronymDef Modify default style to use access text:

```

9725 \renewcommand*\<DefaultNewAcronymDef>{%
9726   \edef\@do@newglossaryentry{%
9727     \noexpand\newglossaryentry{\the\glslabeltok}%
9728     {%
9729       type=\acronymtype,%
9730       name={\the\glsshorttok},%
9731       description={\the\glslongtok},%
9732       descriptionaccess=\relax,%
9733       text={\the\glsshorttok},%

```

```

9734     access={\noexpand\@glo@textaccess},%
9735     sort={\the\glsshorttok},%
9736     short={\the\glsshorttok},%
9737     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9738     shortaccess={\the\glslongtok},%
9739     long={\the\glslongtok},%
9740     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9741     descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9742     first={\noexpand\glslongaccessdisplay
9743           {\the\glslongtok}{\the\glslabeltok}\space
9744           (\noexpand\glsshortaccessdisplay
9745           {\the\glsshorttok}{\the\glslabeltok})},%
9746     plural={\the\glsshorttok\acrpluralsuffix},%
9747     firstplural={\noexpand\glslongpluralaccessdisplay
9748                 {\noexpand\@glo@longpl}{\the\glslabeltok}\space
9749                 (\noexpand\glsshortpluralaccessdisplay
9750                 {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
9751     firstaccess=\relax,
9752     firstpluralaccess=\relax,
9753     textaccess={\noexpand\@glo@shortaccess},%
9754     \the\glskeylisttok
9755 }%
9756 }%
9757 \let\@org@gls@assign@firstpl\gls@assign@firstpl
9758 \let\@org@gls@assign@plural\gls@assign@plural
9759 \let\@org@gls@assign@descplural\gls@assign@descplural
9760 \def\gls@assign@firstpl##1##2{%
9761   \@@gls@expand@field{##1}{firstpl}{##2}%
9762 }%
9763 \def\gls@assign@plural##1##2{%
9764   \@@gls@expand@field{##1}{plural}{##2}%
9765 }%
9766 \def\gls@assign@descplural##1##2{%
9767   \@@gls@expand@field{##1}{descplural}{##2}%
9768 }%
9769 \@do@newglossaryentry
9770 \let\gls@assign@firstpl\@org@gls@assign@firstpl
9771 \let\gls@assign@plural\@org@gls@assign@plural
9772 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
9773 }

```

otnoteNewAcronymDef

```

9774 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
9775   \edef\@do@newglossaryentry{%
9776     \noexpand\newglossaryentry{\the\glslabeltok}%
9777     {%
9778       type=\acronymtype,%
9779       name={\noexpand\acronymfont{\the\glsshorttok}},%
9780       sort={\the\glsshorttok},%

```

```

9781     text={\the\glsshorttok},%
9782     short={\the\glsshorttok},%
9783     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9784     shortaccess={\the\glslongtok},%
9785     long={\the\glslongtok},%
9786     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9787     access={\noexpand\@glo@textaccess},%
9788     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9789     symbol={\the\glslongtok},%
9790     symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9791     firstpluralaccess=\relax,
9792     textaccess={\noexpand\@glo@shortaccess},%
9793     \the\glskeylisttok
9794 }%
9795 }%
9796 \let\@org@gls@assign@firstpl\gls@assign@firstpl
9797 \let\@org@gls@assign@plural\gls@assign@plural
9798 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
9799 \def\gls@assign@firstpl##1##2{%
9800   \@@gls@expand@field{##1}{firstpl}{##2}%
9801 }%
9802 \def\gls@assign@plural##1##2{%
9803   \@@gls@expand@field{##1}{plural}{##2}%
9804 }%
9805 \def\gls@assign@symbolplural##1##2{%
9806   \@@gls@expand@field{##1}{symbolplural}{##2}%
9807 }%
9808 \do@newglossaryentry
9809 \let\gls@assign@plural\@org@gls@assign@plural
9810 \let\gls@assign@firstpl\@org@gls@assign@firstpl
9811 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
9812 }

```

ptionNewAcronymDef

```

9813 \renewcommand*{\DescriptionNewAcronymDef}{%
9814   \edef\@do@newglossaryentry{%
9815     \noexpand\newglossaryentry{\the\glslabeltok}%
9816     {%
9817       type=\acronymtype,%
9818       name={\noexpand
9819         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
9820       access={\noexpand\@glo@textaccess},%
9821       sort={\the\glsshorttok},%
9822       short={\the\glsshorttok},%
9823       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9824       shortaccess={\the\glslongtok},%
9825       long={\the\glslongtok},%
9826       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9827       first={\the\glslongtok},%

```



```

9828     firstaccess=\relax,
9829     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9830     text={\the\glsshorttok},%
9831     textaccess={\the\glslongtok},%
9832     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9833     symbol={\noexpand\@glo@text},%
9834     symbolaccess={\noexpand\@glo@textaccess},%
9835     symbolplural={\noexpand\@glo@plural},%
9836     firstpluralaccess=\relax,
9837     textaccess={\noexpand\@glo@shortaccess},%
9838     \the\glskeylisttok}%
9839 }%
9840 \let\@org@gls@assign@firstpl\gls@assign@firstpl
9841 \let\@org@gls@assign@plural\gls@assign@plural
9842 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
9843 \def\gls@assign@firstpl##1##2{%
9844   \@@gls@expand@field{##1}{firstpl}{##2}%
9845 }%
9846 \def\gls@assign@plural##1##2{%
9847   \@@gls@expand@field{##1}{plural}{##2}%
9848 }%
9849 \def\gls@assign@symbolplural##1##2{%
9850   \@@gls@expand@field{##1}{symbolplural}{##2}%
9851 }%
9852 \do@newglossaryentry
9853 \let\gls@assign@firstpl\@org@gls@assign@firstpl
9854 \let\gls@assign@plural\@org@gls@assign@plural
9855 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
9856 }

```

otnoteNewAcronymDef

```

9857 \renewcommand*{\FootnoteNewAcronymDef}{%
9858   \edef\@do@newglossaryentry{%
9859     \noexpand\newglossaryentry{\the\glslabeltok}%
9860     {%
9861       type=\acronymtype,%
9862       name={\noexpand\acronymfont{\the\glsshorttok}},%
9863       sort={\the\glsshorttok},%
9864       text={\the\glsshorttok},%
9865       textaccess={\the\glslongtok},%
9866       access={\noexpand\@glo@textaccess},%
9867       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9868       short={\the\glsshorttok},%
9869       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9870       long={\the\glslongtok},%
9871       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9872       description={\the\glslongtok},%
9873       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9874       \the\glskeylisttok

```

```

9875 }%
9876 }%
9877 \let\@org@gls@assign@plural\gls@assign@plural
9878 \let\@org@gls@assign@firstpl\gls@assign@firstpl
9879 \let\@org@gls@assign@descplural\gls@assign@descplural
9880 \def\gls@assign@firstpl##1##2{%
9881   \@@gls@expand@field{##1}{firstpl}{##2}%
9882 }%
9883 \def\gls@assign@plural##1##2{%
9884   \@@gls@expand@field{##1}{plural}{##2}%
9885 }%
9886 \def\gls@assign@descplural##1##2{%
9887   \@@gls@expand@field{##1}{descplural}{##2}%
9888 }%
9889 \do@newglossaryentry
9890 \let\gls@assign@plural\@org@gls@assign@plural
9891 \let\gls@assign@firstpl\@org@gls@assign@firstpl
9892 \let\gls@assign@descplural\@org@gls@assign@descplural
9893 }

```

\SmallNewAcronymDef

```

9894 \renewcommand*{\SmallNewAcronymDef}{%
9895   \edef\@do@newglossaryentry{%
9896     \noexpand\newglossaryentry{\the\glslabeltok}%
9897     {%
9898       type=\acronymtype,%
9899       name={\noexpand\acronymfont{\the\glsshorttok}},%
9900       access={\noexpand\@glo@symbolaccess},%
9901       sort={\the\glsshorttok},%
9902       short={\the\glsshorttok},%
9903       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9904       shortaccess={\the\glslongtok},%
9905       long={\the\glslongtok},%
9906       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9907       text={\noexpand\@glo@short},%
9908       textaccess={\noexpand\@glo@shortaccess},%
9909       plural={\noexpand\@glo@shortpl},%
9910       first={\the\glslongtok},%
9911       firstaccess=\relax,%
9912       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9913       description={\noexpand\@glo@first},%
9914       descriptionplural={\noexpand\@glo@firstplural},%
9915       symbol={\the\glsshorttok},%
9916       symbolaccess={\the\glslongtok},%
9917       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9918       \the\glskeylisttok
9919     }%
9920   }%
9921   \let\@org@gls@assign@firstpl\gls@assign@firstpl

```

```

9922 \let\@org@glsgls@assign@plural\glsgls@assign@plural
9923 \let\@org@glsgls@assign@descplural\glsgls@assign@descplural
9924 \let\@org@glsgls@assign@symbolplural\glsgls@assign@symbolplural
9925 \def\glsgls@assign@firstpl##1##2{%
9926   \@@glsgls@expand@field{##1}{firstpl}{##2}%
9927 }%
9928 \def\glsgls@assign@plural##1##2{%
9929   \@@glsgls@expand@field{##1}{plural}{##2}%
9930 }%
9931 \def\glsgls@assign@descplural##1##2{%
9932   \@@glsgls@expand@field{##1}{descplural}{##2}%
9933 }%
9934 \def\glsgls@assign@symbolplural##1##2{%
9935   \@@glsgls@expand@field{##1}{symbolplural}{##2}%
9936 }%
9937 \do@newglossaryentry
9938 \let\glsgls@assign@firstpl\@org@glsgls@assign@firstpl
9939 \let\glsgls@assign@plural\@org@glsgls@assign@plural
9940 \let\glsgls@assign@descplural\@org@glsgls@assign@descplural
9941 \let\glsgls@assign@symbolplural\@org@glsgls@assign@symbolplural
9942 }

```

The following are kept for compatibility with versions before 3.0:

\glsshortaccesskey

```

9943 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

```

\glshortpluralaccesskey

```

9944 \newcommand*{\glshortpluralaccesskey}{\glshortpluralkey access}%

```

\glslongaccesskey

```

9945 \newcommand*{\glslongaccesskey}{\glslongkey access}%

```

\glslongpluralaccesskey

```

9946 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

6.5 Debugging Commands

\showglongnameaccess

```

9947 \newcommand*{\showglongnameaccess}[1]{%
9948   \expandafter\show\csname glo@\glsglsetoklabel{#1}@textaccess\endcsname
9949 }

```

\showglotextaccess

```

9950 \newcommand*{\showglotextaccess}[1]{%
9951   \expandafter\show\csname glo@\glsglsetoklabel{#1}@textaccess\endcsname
9952 }

```

showglopluralaccess

```
9953 \newcommand*{\showglopluralaccess}[1]{%
9954   \expandafter\show\csname glo@\glsdetoklabel{#1}@pluralaccess\endcsname
9955 }
```

\showglofirstaccess

```
9956 \newcommand*{\showglofirstaccess}[1]{%
9957   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstaccess\endcsname
9958 }
```

lofirstpluralaccess

```
9959 \newcommand*{\showglofirstpluralaccess}[1]{%
9960   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpluralaccess\endcsname
9961 }
```

showglosymbolaccess

```
9962 \newcommand*{\showglosymbolaccess}[1]{%
9963   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolaccess\endcsname
9964 }
```

osymbolpluralaccess

```
9965 \newcommand*{\showglosymbolpluralaccess}[1]{%
9966   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolpluralaccess\endcsname
9967 }
```

\showglodescaccess

```
9968 \newcommand*{\showglodescaccess}[1]{%
9969   \expandafter\show\csname glo@\glsdetoklabel{#1}@descaccess\endcsname
9970 }
```

glodescpluralaccess

```
9971 \newcommand*{\showglodescpluralaccess}[1]{%
9972   \expandafter\show\csname glo@\glsdetoklabel{#1}@descpluralaccess\endcsname
9973 }
```

\showgloshortaccess

```
9974 \newcommand*{\showgloshortaccess}[1]{%
9975   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortaccess\endcsname
9976 }
```

loshortpluralaccess

```
9977 \newcommand*{\showgloshortpluralaccess}[1]{%
9978   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortpluralaccess\endcsname
9979 }
```

\showglolongaccess

```
9980 \newcommand*{\showglolongaccess}[1]{%
9981   \expandafter\show\csname glo@\glsdetoklabel{#1}@longaccess\endcsname
9982 }
```

glo\longpluralaccess

```
9983 \newcommand*{\showglo\longpluralaccess}[1]{%
9984   \expandafter\show\csname glo@\glsdetoklabel{#1}@longpluralaccess\endcsname
9985 }
```

7 Multi-Lingual Support

Many thanks to everyone who contributed to the translations both via email and on comp.text.tex.

7.1 Babel Captions

Define captions if multi-lingual support is required, but the package is not loaded.

```
9986 \NeedsTeXFormat{LaTeX2e}
9987 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]
```

English:

```
9988 \@ifundefined{captionsenglish}{\}%
9989   \addto\captionsenglish{%
9990     \renewcommand*{\glossaryname}{Glossary}%
9991     \renewcommand*{\acronymname}{Acronyms}%
9992     \renewcommand*{\entryname}{Notation}%
9993     \renewcommand*{\descriptionname}{Description}%
9994     \renewcommand*{\symbolname}{Symbol}%
9995     \renewcommand*{\pagelistname}{Page List}%
9996     \renewcommand*{\glssymbolsgroupname}{Symbols}%
9997     \renewcommand*{\glsnumbersgroupname}{Numbers}%
9998 }%
9999 }
10000 \@ifundefined{captionsamerican}{\}%
10001   \addto\captionsamerican{%
10002     \renewcommand*{\glossaryname}{Glossary}%
10003     \renewcommand*{\acronymname}{Acronyms}%
10004     \renewcommand*{\entryname}{Notation}%
10005     \renewcommand*{\descriptionname}{Description}%
10006     \renewcommand*{\symbolname}{Symbol}%
10007     \renewcommand*{\pagelistname}{Page List}%
10008     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10009     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10010 }%
10011 }
10012 \@ifundefined{captionsaustralian}{\}%
10013   \addto\captionsaustralian{%
10014     \renewcommand*{\glossaryname}{Glossary}%
10015     \renewcommand*{\acronymname}{Acronyms}%
10016     \renewcommand*{\entryname}{Notation}%
10017     \renewcommand*{\descriptionname}{Description}%
```

```

10018 \renewcommand*{\symbolname}{Symbol}%
10019 \renewcommand*{\pagelistname}{Page List}%
10020 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10021 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10022 }%
10023 }
10024 \@ifundefined{captionsbritish}{-}{%
10025 \addto\captionsbritish{%
10026 \renewcommand*{\glossaryname}{Glossary}%
10027 \renewcommand*{\acronymname}{Acronyms}%
10028 \renewcommand*{\entryname}{Notation}%
10029 \renewcommand*{\descriptionname}{Description}%
10030 \renewcommand*{\symbolname}{Symbol}%
10031 \renewcommand*{\pagelistname}{Page List}%
10032 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10033 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10034 }}%
10035 \@ifundefined{captionscanadian}{-}{%
10036 \addto\captionscanadian{%
10037 \renewcommand*{\glossaryname}{Glossary}%
10038 \renewcommand*{\acronymname}{Acronyms}%
10039 \renewcommand*{\entryname}{Notation}%
10040 \renewcommand*{\descriptionname}{Description}%
10041 \renewcommand*{\symbolname}{Symbol}%
10042 \renewcommand*{\pagelistname}{Page List}%
10043 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10044 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10045 }%
10046 }
10047 \@ifundefined{captionsnewzealand}{-}{%
10048 \addto\captionsnewzealand{%
10049 \renewcommand*{\glossaryname}{Glossary}%
10050 \renewcommand*{\acronymname}{Acronyms}%
10051 \renewcommand*{\entryname}{Notation}%
10052 \renewcommand*{\descriptionname}{Description}%
10053 \renewcommand*{\symbolname}{Symbol}%
10054 \renewcommand*{\pagelistname}{Page List}%
10055 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10056 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10057 }%
10058 }
10059 \@ifundefined{captionsUKenglish}{-}{%
10060 \addto\captionsUKenglish{%
10061 \renewcommand*{\glossaryname}{Glossary}%
10062 \renewcommand*{\acronymname}{Acronyms}%
10063 \renewcommand*{\entryname}{Notation}%
10064 \renewcommand*{\descriptionname}{Description}%
10065 \renewcommand*{\symbolname}{Symbol}%
10066 \renewcommand*{\pagelistname}{Page List}%

```

```

10067 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10068 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10069 }%
10070 }
10071 \@ifundefined{captionsUSenglish}{-}{%
10072 \addto\captionsUSenglish{%
10073 \renewcommand*{\glossaryname}{Glossary}%
10074 \renewcommand*{\acronymname}{Acronyms}%
10075 \renewcommand*{\entryname}{Notation}%
10076 \renewcommand*{\descriptionname}{Description}%
10077 \renewcommand*{\symbolname}{Symbol}%
10078 \renewcommand*{\pagelistname}{Page List}%
10079 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10080 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10081 }%
10082 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

10083 \@ifundefined{captionsgerman}{-}{%
10084 \addto\captionsgerman{%
10085 \renewcommand*{\glossaryname}{Glossar}%
10086 \renewcommand*{\acronymname}{Akronyme}%
10087 \renewcommand*{\entryname}{Bezeichnung}%
10088 \renewcommand*{\descriptionname}{Beschreibung}%
10089 \renewcommand*{\symbolname}{Symbol}%
10090 \renewcommand*{\pagelistname}{Seiten}%
10091 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10092 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10093 }

```

ngerman is identical to German:

```

10094 \@ifundefined{captionsgerman}{-}{%
10095 \addto\captionsgerman{%
10096 \renewcommand*{\glossaryname}{Glossar}%
10097 \renewcommand*{\acronymname}{Akronyme}%
10098 \renewcommand*{\entryname}{Bezeichnung}%
10099 \renewcommand*{\descriptionname}{Beschreibung}%
10100 \renewcommand*{\symbolname}{Symbol}%
10101 \renewcommand*{\pagelistname}{Seiten}%
10102 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10103 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10104 }

```

Italian:

```

10105 \@ifundefined{captionssitalian}{-}{%
10106 \addto\captionssitalian{%
10107 \renewcommand*{\glossaryname}{Glossario}%
10108 \renewcommand*{\acronymname}{Acronimi}%
10109 \renewcommand*{\entryname}{Nomenclatura}%
10110 \renewcommand*{\descriptionname}{Descrizione}%

```

```

10111 \renewcommand*{\symbolname}{Simbolo}%
10112 \renewcommand*{\pagelistname}{Elenco delle pagine}%
10113 \renewcommand*{\glssymbolsgroupname}{Simboli}%
10114 \renewcommand*{\glsnumbersgroupname}{Numeri}}
10115 }

```

Dutch:

```

10116 \@ifundefined{captionsdutch}{}{%
10117 \addto\captionsdutch{%
10118 \renewcommand*{\glossaryname}{Woordenlijst}%
10119 \renewcommand*{\acronymname}{Acroniemen}%
10120 \renewcommand*{\entryname}{Benaming}%
10121 \renewcommand*{\descriptionname}{Beschrijving}%
10122 \renewcommand*{\symbolname}{Symbool}%
10123 \renewcommand*{\pagelistname}{Pagina's}%
10124 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
10125 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
10126 }

```

Spanish:

```

10127 \@ifundefined{captionsspanish}{}{%
10128 \addto\captionsspanish{%
10129 \renewcommand*{\glossaryname}{Glosario}%
10130 \renewcommand*{\acronymname}{Siglas}%
10131 \renewcommand*{\entryname}{Entrada}%
10132 \renewcommand*{\descriptionname}{Descripci'on}%
10133 \renewcommand*{\symbolname}{S'\i mbolo}%
10134 \renewcommand*{\pagelistname}{Lista de p'aginas}%
10135 \renewcommand*{\glssymbolsgroupname}{S'\i mbolos}%
10136 \renewcommand*{\glsnumbersgroupname}{N'umeros}}
10137 }

```

French:

```

10138 \@ifundefined{captionsfrench}{}{%
10139 \addto\captionsfrench{%
10140 \renewcommand*{\glossaryname}{Glossaire}%
10141 \renewcommand*{\acronymname}{Acronymes}%
10142 \renewcommand*{\entryname}{Terme}%
10143 \renewcommand*{\descriptionname}{Description}%
10144 \renewcommand*{\symbolname}{Symbole}%
10145 \renewcommand*{\pagelistname}{Pages}%
10146 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10147 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10148 }
10149 \@ifundefined{captionsfrenchb}{}{%
10150 \addto\captionsfrenchb{%
10151 \renewcommand*{\glossaryname}{Glossaire}%
10152 \renewcommand*{\acronymname}{Acronymes}%
10153 \renewcommand*{\entryname}{Terme}%
10154 \renewcommand*{\descriptionname}{Description}%
10155 \renewcommand*{\symbolname}{Symbole}%

```



```

10156 \renewcommand*{\pagelistname}{Pages}%
10157 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10158 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10159 }
10160 \@ifundefined{captionsfrançais}{\{}{
10161 \addto\captionsfrançais{
10162 \renewcommand*{\glossaryname}{Glossaire}%
10163 \renewcommand*{\acronymname}{Acronymes}%
10164 \renewcommand*{\entryname}{Terme}%
10165 \renewcommand*{\descriptionname}{Description}%
10166 \renewcommand*{\symbolname}{Symbole}%
10167 \renewcommand*{\pagelistname}{Pages}%
10168 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10169 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10170 }

```

Danish:

```

10171 \@ifundefined{captionsdanish}{\{}{
10172 \addto\captionsdanish{
10173 \renewcommand*{\glossaryname}{Ordliste}%
10174 \renewcommand*{\acronymname}{Akronymer}%
10175 \renewcommand*{\entryname}{Symbolforklaring}%
10176 \renewcommand*{\descriptionname}{Beskrivelse}%
10177 \renewcommand*{\symbolname}{Symbol}%
10178 \renewcommand*{\pagelistname}{Side}%
10179 \renewcommand*{\glssymbolsgroupname}{Symboler}%
10180 \renewcommand*{\glsnumbersgroupname}{Tal}}
10181 }

```

Irish:

```

10182 \@ifundefined{captionsirish}{\{}{
10183 \addto\captionsirish{
10184 \renewcommand*{\glossaryname}{Gluais}%
10185 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

10186 \renewcommand*{\entryname}{Ciall}%
10187 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

10188 \renewcommand*{\symbolname}{Comhartha}%
10189 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
10190 \renewcommand*{\pagelistname}{Leathanaigh}%
10191 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}
10192 }

```

Hungarian:

```

10193 \@ifundefined{captionsmagyar}{\{}{
10194 \addto\captionsmagyar{

```

```

10195 \renewcommand*{\glossaryname}{Sz\’ojegyz\’ek}%
10196 \renewcommand*{\acronymname}{Bet\H uszavak}%
10197 \renewcommand*{\entryname}{Kifejez\’es}%
10198 \renewcommand*{\descriptionname}{Magyar\’azat}%
10199 \renewcommand*{\symbolname}{Jel\’ol\’es}%
10200 \renewcommand*{\pagelistname}{Oldalsz\’am}%
10201 \renewcommand*{\glssymbolsgroupname}{Jelek}%
10202 \renewcommand*{\glsnumbersgroupname}{Sz\’amjegyek}%
10203 }
10204 }
10205 \@ifundefined{captionshungarian}{}{%
10206 \addto\captionshungarian{%
10207 \renewcommand*{\glossaryname}{Sz\’ojegyz\’ek}%
10208 \renewcommand*{\acronymname}{Bet\H uszavak}%
10209 \renewcommand*{\entryname}{Kifejez\’es}%
10210 \renewcommand*{\descriptionname}{Magyar\’azat}%
10211 \renewcommand*{\symbolname}{Jel\’ol\’es}%
10212 \renewcommand*{\pagelistname}{Oldalsz\’am}%
10213 \renewcommand*{\glssymbolsgroupname}{Jelek}%
10214 \renewcommand*{\glsnumbersgroupname}{Sz\’amjegyek}%
10215 }
10216 }

```

Polish

```

10217 \@ifundefined{captionspolish}{}{%
10218 \addto\captionspolish{%
10219 \renewcommand*{\glossaryname}{S{l}ownik termin\’ow}%
10220 \renewcommand*{\acronymname}{Skr\’ot}%
10221 \renewcommand*{\entryname}{Termin}%
10222 \renewcommand*{\descriptionname}{Opis}%
10223 \renewcommand*{\symbolname}{Symbol}%
10224 \renewcommand*{\pagelistname}{Strony}%
10225 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10226 \renewcommand*{\glsnumbersgroupname}{Liczby}}
10227 }

```

Brazilian

```

10228 \@ifundefined{captionsbrazil}{}{%
10229 \addto\captionsbrazil{%
10230 \renewcommand*{\glossaryname}{Gloss\’ario}%
10231 \renewcommand*{\acronymname}{Siglas}%
10232 \renewcommand*{\entryname}{Nota\c c\~ao}%
10233 \renewcommand*{\descriptionname}{Descri\c c\~ao}%
10234 \renewcommand*{\symbolname}{S\’imbolo}%
10235 \renewcommand*{\pagelistname}{Lista de P\’aginas}%
10236 \renewcommand*{\glssymbolsgroupname}{S\’imbolos}%
10237 \renewcommand*{\glsnumbersgroupname}{N\’umeros}%
10238 }%
10239 }

```

7.2 Polyglossia Captions

```
10240 \NeedsTeXFormat{LaTeX2e}
10241 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]
```

English:

```
10242 \@ifundefined{captionsenglish}{\}%
10243   \expandafter\toks@\expandafter{\captionsenglish
10244     \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
10245     \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
10246     \renewcommand*{\entryname}{\textenglish{Notation}}%
10247     \renewcommand*{\descriptionname}{\textenglish{Description}}%
10248     \renewcommand*{\symbolname}{\textenglish{Symbol}}%
10249     \renewcommand*{\pagelistname}{\textenglish{Page List}}%
10250     \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
10251     \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
10252   }%
10253   \edef\captionsenglish{\the\toks@}%
10254 }
```

German:

```
10255 \@ifundefined{captionsgerman}{\}%
10256   \expandafter\toks@\expandafter{\captionsgerman
10257     \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
10258     \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
10259     \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
10260     \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
10261     \renewcommand*{\symbolname}{\textgerman{Symbol}}%
10262     \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
10263     \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
10264     \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
10265   }%
10266   \edef\captionsgerman{\the\toks@}%
10267 }
```

Italian:

```
10268 \@ifundefined{captionsitalian}{\}%
10269   \expandafter\toks@\expandafter{\captionsitalian
10270     \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
10271     \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
10272     \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
10273     \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
10274     \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
10275     \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
10276     \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
10277     \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
10278   }%
10279   \edef\captionsitalian{\the\toks@}%
10280 }
```

Dutch:

```
10281 \@ifundefined{captionsdutch}{\}%
```

```

10282 \expandafter\toks@\expandafter{\captionsdutch
10283 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}}%
10284 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}}%
10285 \renewcommand*{\entryname}{\textdutch{Benaming}}}%
10286 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}}%
10287 \renewcommand*{\symbolname}{\textdutch{Symbool}}}%
10288 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}}%
10289 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}}%
10290 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}}%
10291 }%
10292 \edef\captionsdutch{\the\toks@}%
10293 }

```

Spanish:

```

10294 \@ifundefined{captionsspanish}{}{%
10295 \expandafter\toks@\expandafter{\captionsspanish
10296 \renewcommand*{\glossaryname}{\textspanish{Glosario}}}%
10297 \renewcommand*{\acronymname}{\textspanish{Siglas}}}%
10298 \renewcommand*{\entryname}{\textspanish{Entrada}}}%
10299 \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}}%
10300 \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}}%
10301 \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}}%
10302 \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}}%
10303 \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}}%
10304 }%
10305 \edef\captionsspanish{\the\toks@}%
10306 }

```

French:

```

10307 \@ifundefined{captionsfrench}{}{%
10308 \expandafter\toks@\expandafter{\captionsfrench
10309 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}}%
10310 \renewcommand*{\acronymname}{\textfrench{Acronymes}}}%
10311 \renewcommand*{\entryname}{\textfrench{Terme}}}%
10312 \renewcommand*{\descriptionname}{\textfrench{Description}}}%
10313 \renewcommand*{\symbolname}{\textfrench{Symbole}}}%
10314 \renewcommand*{\pagelistname}{\textfrench{Pages}}}%
10315 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}}%
10316 \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}}%
10317 }%
10318 \edef\captionsfrench{\the\toks@}%
10319 }

```

Danish:

```

10320 \@ifundefined{captionsdanish}{}{%
10321 \expandafter\toks@\expandafter{\captionsdanish
10322 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}}%
10323 \renewcommand*{\acronymname}{\textdanish{Akronymer}}}%
10324 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}}%
10325 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}}%
10326 \renewcommand*{\symbolname}{\textdanish{Symbol}}}%

```

```

10327 \renewcommand*{\pagelistname}{\textdanish{Side}}%
10328 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
10329 \renewcommand*{\glsnumbersgroupname}{\textdanish{Tal}}%
10330 }%
10331 \edef\captionsdanish{\the\toks@}%
10332 }

Irish:
10333 \@ifundefined{captionsirish}{}{%
10334 \expandafter\toks@\expandafter{\captionsirish
10335 \renewcommand*{\glossaryname}{\textirish{Gluais}}%
10336 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%
10337 \renewcommand*{\entryname}{\textirish{Ciall}}%
10338 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}%
10339 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
10340 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha'\{i}}}%
10341 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%
10342 \renewcommand*{\glsnumbersgroupname}{\textirish{Uimhreacha}}%
10343 }%
10344 \edef\captionsirish{\the\toks@}%
10345 }

Hungarian:
10346 \@ifundefined{captionsmagyar}{}{%
10347 \expandafter\toks@\expandafter{\captionsmagyar
10348 \renewcommand*{\glossaryname}{\textmagyar{Sz\'ojegy\'ek}}%
10349 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}%
10350 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}%
10351 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'azat}}%
10352 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}%
10353 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}%
10354 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}%
10355 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\'amjegyek}}%
10356 }%
10357 \edef\captionsmagyar{\the\toks@}%
10358 }

Polish
10359 \@ifundefined{captionspolish}{}{%
10360 \expandafter\toks@\expandafter{\captionspolish
10361 \renewcommand*{\glossaryname}{\textpolish{S\lownik termin\'ow}}%
10362 \renewcommand*{\acronymname}{\textpolish{Skr\'ot}}%
10363 \renewcommand*{\entryname}{\textpolish{Termin}}%
10364 \renewcommand*{\descriptionname}{\textpolish{Opis}}%
10365 \renewcommand*{\symbolname}{\textpolish{Symbol}}%
10366 \renewcommand*{\pagelistname}{\textpolish{Strony}}%
10367 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}%
10368 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}%
10369 }%
10370 \edef\captionspolish{\the\toks@}%
10371 }

```

Portugues

```
10372 \@ifundefined{captionsportuges}{}{%
10373   \expandafter\toks@ \expandafter{\captionsportuges
10374     \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}%
10375     \renewcommand*{\acronymname}{\textportuges{Siglas}}%
10376     \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
10377     \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
10378     \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}%
10379     \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}%
10380     \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}%
10381     \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}%
10382   }%
10383   \edef\captionsportuges{\the\toks@}%
10384 }
```

7.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```
10385 \ProvidesDictionary{glossaries-dictionary}{Brazilian}
```

Provide Brazilian translations:

```
10386 \providetranslation{Glossary}{Gloss\'ario}
10387 \providetranslation{Acronyms}{Siglas}
10388 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
10389 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
10390 \providetranslation{Symbol (glossaries)}{S\'imbolo}
10391 \providetranslation{Page List (glossaries)}{Lista de P\'aginas}
10392 \providetranslation{Symbols (glossaries)}{S\'imbolos}
10393 \providetranslation{Numbers (glossaries)}{N\'umeros}
```

7.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```
10394 \ProvidesDictionary{glossaries-dictionary}{Danish}
```

Provide Danish translations:

```
10395 \providetranslation{Glossary}{Ordliste}
10396 \providetranslation{Acronyms}{Akronymer}
10397 \providetranslation{Notation (glossaries)}{Symbolforklaring}
10398 \providetranslation{Description (glossaries)}{Beskrivelse}
10399 \providetranslation{Symbol (glossaries)}{Symbol}
10400 \providetranslation{Page List (glossaries)}{Side}
10401 \providetranslation{Symbols (glossaries)}{Symboler}
10402 \providetranslation{Numbers (glossaries)}{Tal}
```

7.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
10403 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
10404 \providetranslation{Glossary}{Woordenlijst}
10405 \providetranslation{Acronyms}{Acroniemen}
10406 \providetranslation{Notation (glossaries)}{Benaming}
10407 \providetranslation{Description (glossaries)}{Beschrijving}
10408 \providetranslation{Symbol (glossaries)}{Symbool}
10409 \providetranslation{Page List (glossaries)}{Pagina's}
10410 \providetranslation{Symbols (glossaries)}{Symbolen}
10411 \providetranslation{Numbers (glossaries)}{Cijfers}
```

7.6 English Dictionary

This is a dictionary file provided for use with the package.

```
10412 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
10413 \providetranslation{Glossary}{Glossary}
10414 \providetranslation{Acronyms}{Acronyms}
10415 \providetranslation{Notation (glossaries)}{Notation}
10416 \providetranslation{Description (glossaries)}{Description}
10417 \providetranslation{Symbol (glossaries)}{Symbol}
10418 \providetranslation{Page List (glossaries)}{Page List}
10419 \providetranslation{Symbols (glossaries)}{Symbols}
10420 \providetranslation{Numbers (glossaries)}{Numbers}
```

7.7 French Dictionary

This is a dictionary file provided for use with the package.

```
10421 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
10422 \providetranslation{Glossary}{Glossaire}
10423 \providetranslation{Acronyms}{Acronymes}
10424 \providetranslation{Notation (glossaries)}{Terme}
10425 \providetranslation{Description (glossaries)}{Description}
10426 \providetranslation{Symbol (glossaries)}{Symbole}
10427 \providetranslation{Page List (glossaries)}{Pages}
10428 \providetranslation{Symbols (glossaries)}{Symboles}
10429 \providetranslation{Numbers (glossaries)}{Nombres}
```

7.8 German Dictionary

This is a dictionary file provided for use with the package.

```
10430 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
10431 \providetranslation{Glossary}{Glossar}
10432 \providetranslation{Acronyms}{Akronyme}
```

```

10433 \providetranslation{Notation (glossaries)}{Bezeichnung}
10434 \providetranslation{Description (glossaries)}{Beschreibung}
10435 \providetranslation{Symbol (glossaries)}{Symbol}
10436 \providetranslation{Page List (glossaries)}{Seiten}
10437 \providetranslation{Symbols (glossaries)}{Symbole}
10438 \providetranslation{Numbers (glossaries)}{Zahlen}

```

7.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```

10439 \ProvidesDictionary{glossaries-dictionary}{Irish}

```

Provide Irish translations:

```

10440 \providetranslation{Glossary}{Gluais}
10441 \providetranslation{Acronyms}{Acrainmneacha}
10442 \providetranslation{Notation (glossaries)}{Ciall}
10443 \providetranslation{Description (glossaries)}{Tuairisc}
10444 \providetranslation{Symbol (glossaries)}{Comhartha}
10445 \providetranslation{Page List (glossaries)}{Leathanaigh}
10446 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
10447 \providetranslation{Numbers (glossaries)}{Uimhreacha}

```

7.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```

10448 \ProvidesDictionary{glossaries-dictionary}{Italian}

```

Provide Italian translations:

```

10449 \providetranslation{Glossary}{Glossario}
10450 \providetranslation{Acronyms}{Acronimi}
10451 \providetranslation{Notation (glossaries)}{Nomenclatura}
10452 \providetranslation{Description (glossaries)}{Descrizione}
10453 \providetranslation{Symbol (glossaries)}{Simbolo}
10454 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
10455 \providetranslation{Symbols (glossaries)}{Simboli}
10456 \providetranslation{Numbers (glossaries)}{Numeri}

```

7.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```

10457 \ProvidesDictionary{glossaries-dictionary}{Magyar}

```

Provide translations:

```

10458 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
10459 \providetranslation{Acronyms}{Bet\H uszavak}
10460 \providetranslation{Notation (glossaries)}{Kifejez\'es}
10461 \providetranslation{Description (glossaries)}{Magyar\'azat}
10462 \providetranslation{Symbol (glossaries)}{Jel\'ol\'es}
10463 \providetranslation{Page List (glossaries)}{Oldalsz\'am}

```



```
10464 \providetranslation{Symbols (glossaries)}{Jelek}
10465 \providetranslation{Numbers (glossaries)}{Sz'amjegyek}
```

7.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
10466 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```
10467 \providetranslation{Glossary}{S{l}ownik termin'ow}
10468 \providetranslation{Acronyms}{Skr'ot}
10469 \providetranslation{Notation (glossaries)}{Termin}
10470 \providetranslation{Description (glossaries)}{Opis}
10471 \providetranslation{Symbol (glossaries)}{Symbol}
10472 \providetranslation{Page List (glossaries)}{Strony}
10473 \providetranslation{Symbols (glossaries)}{Symbole}
10474 \providetranslation{Numbers (glossaries)}{Liczby}
```

7.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```
10475 \ProvidesDictionary{glossaries-dictionary}{Serbian}
10476 \providetranslation{Glossary}{Mali re'v cnik}
10477 \providetranslation{Acronyms}{Skra' cenice}
10478 \providetranslation{Notation (glossaries)}{Oznaka}
10479 \providetranslation{Description (glossaries)}{Opis}
10480 \providetranslation{Symbol (glossaries)}{Simbol}
10481 \providetranslation{Page List (glossaries)}{Stranica}
10482 \providetranslation{Symbols (glossaries)}{Simboli}
10483 \providetranslation{Numbers (glossaries)}{Brojevi}
```

7.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```
10484 \ProvidesDictionary{glossaries-dictionary}{Spanish}
```

Provide Spanish translations:

```
10485 \providetranslation{Glossary}{Glosario}
10486 \providetranslation{Acronyms}{Siglas}
10487 \providetranslation{Notation (glossaries)}{Entrada}
10488 \providetranslation{Description (glossaries)}{Descripci'on}
10489 \providetranslation{Symbol (glossaries)}{S'\i mbolo}
10490 \providetranslation{Page List (glossaries)}{Lista de p'aginas}
10491 \providetranslation{Symbols (glossaries)}{S'\i mbolos}
10492 \providetranslation{Numbers (glossaries)}{N'umeros}
```

Glossary

`makeindex` An indexing application. 10, 21

`xindy` An flexible indexing application with multilingual support written in Perl. 10, 21

Change History

- 1.01
 - General: Added range facility in format key 90
 - `\writeist`: Added spaces after `\delimN` and `\delimR` in ist file 141
- 1.03
 - `\makefirstuc`: changed 'protected@edef to 'def 223
- 1.04
 - General: Added `\glstextformat` 76
- 1.05
 - `\glossarysection`: added `\@mkboth to \glossarysection` 32
 - `\gls@defglossaryentry`: Changed the default value of the sort key to just the value of the name key 66
 - `\glsmakefirstuc`: new 224
- 1.06
 - General: now requires `etoolbox` . 222
 - `\capitalisewords`: new 224
 - `\xcapitalisewords`: new 225
- 1.07
 - `\@gls@link`: fixed bug caused by `\theglsentrycounter` setting the page number too soon 88
 - `\glsadd`: fixed bug caused by `\theglsentrycounter` setting the page number too soon 139
- 1.08
 - General: Added babel support ... 27
 - `\capitalisewords`: made robust 224
- `listgroup`: changed `listgroup` style to use `\glsgetgrouptitle` 230
 - `altlistgroup`: changed `altlistgroup` style to use `\glsgetgrouptitle` 231
 - `\makefirstuc`: made robust ... 223
- 1.1
 - `\@glossarysection`: numbered sections and auto label added 34
 - `\@gls@tmpb`: changed `\toksdef` to `\newtoks` 92
 - `\@gls@toc`: numberline added .. 35
 - `\@p@glossarysection`: numbered sections and auto label added 34
 - General: Added support for translator package 28
 - `amsgen` now loaded (`\new@ifnextchar` needed) 4
 - `translate`: `translate` option added 19
 - `\setglossarysection`: new ... 34
 - `numberedsection`: numbered-section package option added . 6
 - `numberline`: `numberline` option added 5
- 1.12
 - `\@GLSpl`: now uses `\glsentrydescplural` and `\glsentrysymbolplural` instead of `\glsentrydesc` and `\glsentrysymbol` 105
 - `\@Glspl`: now uses `\glsentrydescplural` and `\glsentrysymbolplural` instead of `\glsentrydesc` and `\glsentrysymbol` 104

General: added check for \hypertarget separate to \hyperlink (memoir de- fines \hyperlink but not \hypertarget) 98	\glsautoprefix:new 6
descriptionplural:new 53	\glsnavhyperlink: changed 'edef to 'protected@edef ... 225
\gls@defglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended) 66	\glsnavhypertarget: added write to aux file 225
descriptionplural support added 66	\glsnavigation: changed to only use labels for groups that are present 226
symbolplural support added .. 66	1.15
\Glsentrydescplural:New .. 133	\@gls@link: added \glslabel . 88
\glsentrydescplural:New .. 133	General: Added \glssettoctitle 28
\Glsentrysymbolplural:New 134	\gls@defglossaryentry: check for \@glo@first in descrip- tion 69
\glsentrysymbolplural:New 134	check for \@glo@text in sym- bol 70
\SetDescriptionFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 196	\gls@hypergroup: new . 226
\SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 202	\glsnavhypertarget: added check if rerun required 225
symbolplural: new 54	\glssettoctitle: new 26
1.13	\printglossary: changed the way the TOC title is set 157
General: Add Polish support 330, 333 fixed bug that ignored 3rd pa- rameter 108–117	1.16
\ACRfullpl:new 178	\@GLS@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 102
\Acrfullpl:new 178	\@GLSpl: Test glossary type is \acronymtype in addition to checking if footnote option has been used 105
\acrfullpl:new 177	\@GLs@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 101
\acrpluralsuffix:New 174	\@GLspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 104
\gls@defglossaryentry: Changed default first value .. 66	\@gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 99
Changed default firstplural value 66	\@glsdisp: Test glossary type is \acronymtype in addition to checking if footnote option has been used 107
Removed restriction on only using \newglossaryentry in the preamble 71	
\newacronym: Removed re- striction on only using \newacronym in the preamble 174	
1.14	
\@gls@hypergroup: new 226	
General: added nonnumberlist key to \printglossary 161	
added numberedsection key to \printglossary 161	
\firstacronymfont: new 179	

\@glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	103	\ifglxindy:new	21
\@glstarget: raised the hyper- target so the target text doesn't scroll off the top of the page .	98	\istfilename: added xindy sup- port	30
\gls@defglossaryentry: Changed def to let	66	\newglossarystyle: made \newglossarystyle long .	170
1.17		\nopostdesc:new	29
\@do@wrglossary:new	152	nonumberlist:new	55
\@do@seeglossary:new	155	\printglossary: added check to determine if \printglossary is already defined	157
\@glo@storeentry:new	72	added print language to aux file	157
\@glossary: changed defini- tion to use \index instead of \@index	151	order: order package option added	21
\@glsdefaultplural:new	57	\writeist: added xindy support	141
\@glsdefaultsort:new	57	1.18	
\@glshypernumber:new	171	\@gls@loadlist:new	8
\@glsnoname:new	57	\@gls@loadlong:new	8
\@glsnonextpages:new	161	\@gls@loadsuper:new	8
\@wrglossary: modified to allow for xindy support	151	\@gls@loadtree:new	8
General: added Brazilian dictio- nary	334	\gls@defglossaryentry: Changed default value of sort to \@glsdefaultsort	66
Added Brazilian support	330	moved sort sanitization to \newglossaryentry	70
added xindy support	21	\glstarget:new	164
parent:new	55	\oldacronym:new	174
see:new	55	nolist:new	8
\gls@defglossaryentry: added nonumberlist key	66	nolong:new	8
added parent key	66	sort: moved sanitization to \newglossaryentry	53
added see key	66	nostyles:new	8
Stored main part of entry format when entry is defined	70	nosuper:new	8
\gls@suffiF:new	31	notree:new	8
\gls@suffiFF:new	31	1.19	
\glshyperlink:new	139	\glsclearpage:new	35
\glshypernumber: modified to allow material to be attached to location	171	\glsdisp:new	106
\glsnavhyperlink: replaced 'hy- perlink to '@glslink	225	\SetDescriptionAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	200
\glsnavhypertarget: replaced 'hypertarget to '@glstarget .	225	\SetDescriptionFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	196
\glssee:new	155	\SetFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	203
\glsseeformat:new	155		
\glsSetSuffiF:new	31		
\glsSetSuffiFF:new	31		

\SetSmallAcronymStyle:		\@gls@link: added \leavevmode	
changed \acronymfont to	 88	
use \textsmaller instead of		Moved entry existence check to	
\smaller		avoid duplicate code	
206		88	
1.2	General: fixed bug in ngerman	\@glsdisp: Added check for hy-	
	captions	perfirst	107
327		\@glspl@: Added check for hyper-	
2.01		first	103
	\@gls@link: moved \@do@wrglossary	\gls glossarymark: Added check	
	before term is displayed to pre-	to see if it's already defined ..	33
	vent unwanted whatsit	hyperfirst: new	20
	88		
	\forall glossaries: replaced	2.04	
	\ifthenelse with \ifx	\@GLS@: Changed test to check if	
	45	glossary type has been identi-	
	\forall gloss entries: replaced	fied as a list of acronyms ...	102
	\ifthenelse with \ifx	\@GLSpl: Changed test to check if	
	45	glossary type has been identi-	
	\glsdefmain: new	fied as a list of acronyms ...	105
	12	\@GLS@: Changed test to check if	
	\glsdescwidth: changed	glossary type has been identi-	
	\linewidth to \hsize . 233, 248	fied as a list of acronyms ...	101
	\glslistdottedwidth: changed	\@Glspl@: Changed test to check	
	\linewidth to \hsize	if glossary type has been iden-	
	232	tified as a list of acronyms ..	104
	\gls pagelistwidth: changed	\@glossaryentryfield: new ..	71
	\linewidth to \hsize . 233, 248	\@glossarysubentryfield:	
	nomain: added nomain package	new	71
	option	\@gls@: Changed test to check if	
	13	glossary type has been identi-	
	\writeist: removed item_02 - no	fied as a list of acronyms	99
	such makeindex key	\@glsacronymlists: new	14
	146	\@glsdisp: Changed test to check	
2.02		if glossary type has been iden-	
	General: Changed Brazil to Brazil-	tified as a list of acronyms ..	107
	ian	\@glspl@: Changed test to check	
	334	if glossary type has been iden-	
	false will prevent automatic	tified as a list of acronyms ..	103
	loading of translator package	\@newglossaryentryposthook:	
	25	new	71
	\glossarysection: changed	\@newglossaryentryprehook:	
	\@mkboth to \glossarymark	new	71
	32	acronymlists: new	15
	\gls glossarymark: New	\DeclareAcronymList: new ...	14
	33	\DefineAcronymSynonyms: new	191
	\printglossary: suppressed	\gls@def glossaryentry: added	
	warning globally rather than	user1-6 keys	66
	locally	\glsadd: fixed bug that ignored	
	160	counter	139
2.03		\Glsentryuseri: new	135
	\@GLS@: Added check for hyper-		
	first		
	102		
	\@GLSpl: Added check for hyper-		
	first		
	105		
	\@GLS@: Added check for hyper-		
	first		
	101		
	\@Glspl@: Added check for hyper-		
	first		
	104		
	\@gls@: Added check for hyper-		
	first		
	99		
	\@gls@link: new		
	87		

\glsentryuseri:new	135	2.07	
\Glsentryuserii:new	135		General: glssadd format key
\glsentryuserii:new	135		stored in \@glsnumberformat
\Glsentryuseriii:new	136		(was mistakenly stored in
\glsentryuseriii:new	135		\@glo@format) 139
\Glsentryuseriv:new	136	3.0	
\glsentryuseriv:new	136		\@@do@wrglossary: added check
\Glsentryuseriv:new	136		for hyper location prefix ... 153
\Glsentryuseriv:new	136		modified to use new format .. 152
\glsentryuseriv:new	136		\@@glossarysec: replaced
\Glsentryuseriv:new	136		\@ifundefined with
\glsentryuseriv:new	136		\ifcsundef 5
\newglossary: added check to			\@do@seeglossary: Sanitize and
determine if \gls@<type>@display			escape cross-referencing in-
and \gls@<type>@displayfirst			formation 155
have been defined.	51		\@gls@counterwithin: new 9
\SetAcronymLists:new	15		\@gls@ifinlist:new 36
\SetDefaultAcronymDisplayStyle:			\@gls@link: added \@gls@saveentrycounter
new	192	 88
\SetDefaultAcronymStyle:			added \@gls@setsort 88
new	193		\@gls@saveentrycounter: new 89
\SetDescriptionAcronymDisplayStyle:			\@gls@setupsort@def: new ... 11
new	198		\@gls@setupsort@standard:
\SetDescriptionDUAAcronymDisplayStyle:			new 10
new	197		\@gls@setupsort@use: new ... 11
\SetDescriptionFootnoteAcronymDisplayStyle:			\@gls@xdy@locationlist: new 39
new	194		\@gls@link: replaced \@ifundefined
\SetDUADisplayStyle: new ..	206		with \ifcsundef 97
\SetFootnoteAcronymDisplayStyle:			\@gls@nextpages: new 161
new	201		\@makeglossary: Added check
\SetSmallAcronymDisplayStyle:			for savewrites 147
new	203		\@set@glo@numformat: added
2.05			4th argument 90
\@glsdisp: Added closing brace.			\@wrglossary: modified to take
Patch provided by Sergiu			into account savewrites 151
Dotenco	106		\@xdy@attributelist: new 36
Removed spurious brace. Patch			General: added prefix to hyperlink
provided by Sergiu Dotenco	107	 172
\writeist: Added \string be-			etoolbox now loaded 4
fore opening and closing			replaced \@ifundefined with
braces. Patch provided by			\ifcsundef 25, 86, 160
Segiu Dotenco	146		\acrfootnote: new 194
2.06			\ACRfull: added starred version 177
\altnewglossary: new	52		\Acrfull: added starred version 176
\CustomAcronymFields: new .	208		\acrfull: added starred version 175
\CustomNewAcronymDef: new .	209		\ACRfullpl: added starred ver-
\SetCustomDisplayStyle: new	208		sion 178
\SetCustomStyle: new	209		\Acrfullpl: added starred ver-
			sion 178

<code>\acrfullpl</code> : added starred version	177	<code>\glentrytext</code>	139
<code>\acrlinkfootnote</code> : new	194	<code>\glshypernumber</code> : replaced	
<code>\acrnolinkfootnote</code> : new ...	194	<code>\@ifundefined</code> with	
<code>\addglossarytocaptions</code> : replaced		<code>\ifcsundef</code>	171
<code>\@ifundefined</code> with		<code>\glsnumberformat</code> : replaced	
<code>\ifcsundef</code>	27	<code>\@ifundefined</code> with	
<code>savewrites</code> : new	22	<code>\ifcsundef</code>	31
<code>see</code> : added <code>\@gls@seeautonumberlist</code>		<code>\glsrefentry</code> : new	163
.....	55	<code>\glsresetsubentrycounter</code> :	
<code>seeautonumberlist</code> : new	7	new	162
<code>\glossarysection</code> : replaced		<code>\glsseeitem</code> : hyperlink uses	
<code>\@ifundefined</code> with		<code>\glsseeitemformat</code> instead	
<code>\ifcsundef</code>	32	of <code>\glentryname</code>	156
<code>\glossarystyle</code> : replaced		<code>\glsseeitemformat</code> : new	156
<code>\@ifundefined</code> with		<code>\glsortnumberfmt</code> : new	10
<code>\ifcsundef</code>	169	<code>\glsstepentry</code> : new	163
<code>\gls@codepage</code> : replaced		<code>\glsstepsubentry</code> : new	163
<code>\@ifundefined</code> with		<code>\glsesubentrycounterlabel</code> :	
<code>\ifcsundef</code>	22	new	164
<code>\gls@defglossaryentry</code> : added		<code>\glsesubentryitem</code> : new	164
<code>\@gls@defsort</code>	70	<code>theglossary</code> : replaced <code>\@ifundefined</code>	
added short and long keys	66	with <code>\ifcsundef</code>	164
replaced <code>\@ifundefined</code> with		<code>short</code> : new	56
<code>\ifcsundef</code>	67	<code>shortplural</code> : new	56
<code>\gls@doclearpage</code> : replaced		<code>\ifglossaryexists</code> : re-	
<code>\@ifundefined</code> with		placed <code>\@ifundefined</code> with	
<code>\ifcsundef</code>	35	<code>\ifcsundef</code>	45
<code>\glsadd</code> : added <code>\@gls@saveentrycounter</code>		<code>\ifglentryexists</code> : re-	
.....	140	placed <code>\@ifundefined</code> with	
<code>\GlsAddXdyCounters</code> : new	36	<code>\ifcsundef</code>	46
<code>\glentrycounterlabel</code> : new	163	<code>\istfile</code> : deprecated	150
<code>\glentryitem</code> : new	164	<code>glossaryentry</code> : new	162
<code>\Glsentrylong</code> : new	137	<code>glossarysubentry</code> : new	162
<code>\glentrylong</code> : new	137	<code>\newglossary</code> : added <code>\@gls@defsortcount</code>	
<code>\Glsentrylongpl</code> : new	137	52
<code>\glentrylongpl</code> : new	137	replaced <code>\@ifundefined</code> with	
<code>\Glsentryshort</code> : new	136	<code>\ifcsundef</code>	51
<code>\glentryshort</code> : new	136	<code>\newglossaryentry</code> : replaced	
<code>\Glsentryshortpl</code> : new	137	<code>\DeclareRobustCommand</code>	
<code>\glentryshortpl</code> : new	137	with <code>\newrobustcmd</code>	59
<code>\glsgetgrouptitle</code> : re-		<code>\newglossarystyle</code> : re-	
placed <code>\@ifundefined</code> with		placed <code>\@ifundefined</code> with	
<code>\ifcsundef</code>	168	<code>\ifcsundef</code>	170
<code>\glsglossarymark</code> : replaced		<code>entrycounter</code> : new	9
<code>\@ifundefined</code> with		<code>entrycounterwithin</code> : new	9
<code>\ifcsundef</code>	33	<code>\oldacronym</code> : replaced <code>\@ifundefined</code>	
<code>\glshyperlink</code> : changed de-		with <code>\ifcsundef</code>	174
fault from <code>\glentryname</code> to		<code>compatible-2.07</code> : compatible-	
		2.07 option added	22

long:new	56	\writeist: added xindy-only	
longplural:new	57	macro definitions to glossary	
nonumberlist: now boolean ...	55	open tag	143
sort:new	9	modified to support new for-	
counter: replaced \@ifundefined		mat	141
with \ifcsundef	55		
\printglossary: added		\@glswritefiles: added check	
\currentglossary	158	for empty glossaries	150
added \glsnextpages	158	General: made robust	101
make toctitle default to title ..	158	\ACRfull: made robust	176
replaced \@ifundefined with		\Acrfull: made robust	176
\ifcsundef	157, 159	\acrfull: made robust	175
\SetDescriptionFootnoteAcronymDisplayStyle:		\acrfullformat: removed	
expanded options link op-		\acronymfont as it should al-	
tions	194	ready be set in the second ar-	
\setentrycounter: added op-		gument.	176
tional argument	169	\ACRfullpl: made robust	178
\showacronymlists: new	214	\Acrfullpl: made robust	178
\showglocounter: new	211	\acrfullpl: made robust	177
\showglodesc: new	213	\ACRlong: made robust	129
\showglodescplural: new ...	213	\Acrlong: made robust	128
\showglofirst: new	211	\acrlong: made robust	127
\showglofirstpl: new	211	\ACRlongpl: made robust	131
\showgloflag: new	214	\Acrlongpl: made robust	130
\showgloindex: new	214	\acrlongpl: made robust	130
\showglolevel: new	210	\ACRshort: made robust	125
\showglongame: new	213	\Acrshort: made robust	124
\showgloparent: new	210	\acrshort: made robust	123
\showgloplural: new	211	\ACRshortpl: made robust	127
\showglosort: new	213	\Acrshortpl: made robust	126
\showglossaries: new	214	\acrshortpl: made robust	125
\showglossarycounter: new .	215	\Gls: made robust	100
\showglossaryentries: new .	215	\glsadd: made robust	139
\showglossaryin: new	215	\glsaddall: made robust	140
\showglossaryout: new	215	\GLSdesc: made robust	113
\showglossarytitle: new ...	215	\Glsdesc: made robust	113
\showglosymbol: new	213	\glsdesc: made robust	113
\showglosymbolplural: new .	213	\GLSdescplural: made robust .	115
\showglotext: new	211	\Glsdescplural: made robust .	114
\showglotype: new	211	\glsdescplural: made robust .	114
\showglouserii: new	212	\glsfirst: made robust	108
\showglouseriii: new	212	\GLSfirstplural: made robust	111
\showglouseriv: new	212	\Glsfirstplural: made robust	111
\showglouserv: new	212	\glsfirstplural: made robust	111
\showglouservi: new	212	\glslink: made robust	87
subentrycounter: new	9	\GLSname: made robust	112
		\Glsname: made robust	112
		\glsname: made robust	112
		\GLSpl: made robust	105

\Glspl: made robust	104	General: added check for polyglos-	
\glspl: made robust	102	sia	25
\GLSplural: made robust	110	reversed order of package check	29
\GLSsymbol: made robust	116	savenumberlist: new	7
\Glsymbol: made robust	115	ucmark: new	9
\glssymbol: made robust	115	\gls@defglossaryentry: added	
\GLSsymbolplural: made robust		numberlist element	69
.....	117	\gls@save@numberlist: new ..	156
\Glsymbolplural: made robust		\glsdisplaynumberlist: new	138
.....	116	\glsentrycounter: set default	
\glssymbolplural: made robust		value	89
.....	116	\Glsentryfull: fixed bug (re-	
\Glstext: made robust	108	placed \glsentryshortpl	
\glstext: made robust	107	with \glsentryshort)	137
\GLSuseri: made robust	118	\glsentryfullpl: fixed bug (re-	
\Glsuseri: made robust	117	placed \glsentryshort with	
\glssuseri: made robust	117	\glsentryshortpl)	137
\GLSuserii: made robust	119	\glsentrynumberlist: new ..	138
\Glsuserii: made robust	118	\glsmoveentry: new	71
\glssuserii: made robust	118	\glsnumlistlastsep: new ...	139
\GLSuseriii: made robust	120	\glsnumlistsep: new	139
\Glsuseriii: made robust	120	\glsresetsubentrycounter:	
\glssuseriii: made robust	119	new	163
\GLSuseriv: made robust	121	\ifglshaschildren: new	47
\Glsuseriv: made robust	121	\ifglshasparent: new	48
\glssuseriv: made robust	120	\makeglossaries: added list	
\GLSuserv: made robust	122	parser	149
\Glsuserv: made robust	122	indexonlyfirst: new	20
\glssuserv: made robust	121	\printglossary: add a way to	
\GLSuservi: made robust	123	fetch current entry label ...	158
\Glsuservi: made robust	123	\renewglossarystyle: new ..	170
\glssuservi: made robust	122	\showglossaryentries: fixed	
		misspelt command	215
3.02		\SmallNewAcronymDef: fixed	
\@do@wrglossary: changed		broken short and long plural	204
\@glslocref to \theglsentrycounter			
.....	154	3.03	
\@do@wrglossary: changed		\@gls@sanitizesort: new	17
\@do@wr@glossary to test for		\@gls@setupsort@standard:	
indexonlyfirst option; put old		used \@gls@sanitizesort ..	10
\@do@wr@glossary code into		General: allow title to set toctitle	160
\@do@wrglossary	152	\glsinlinedescformat: new ..	229
\@gls@missingnumberlist:		\glsinlineemptydescformat:	
new	57	new	229
\@glswritefiles: added check		\glsinlinenameformat: new ..	229
for existence of token in case		\glsinlinepostchild: new ..	229
\makeglossaries has been		\glsinlinesubdescformat:	
omitted	150	new	229
\@wrglossary: added check for		\glsinlinesubnameformat:	
glossary file defined	151	new	229

<code>\glspostinline:</code> replaced “.” with <code>\glspostdescription</code> 229	General: Added check for doc package 4
<code>altnogragged4col:</code> added check for <code>glsnogroupskip</code> .. 243	added <code>datatool-base</code> as a re- quired package 4
<code>altsuperragged4col:</code> added check for <code>glsnogroupskip</code> .. 259	added local key 87
<code>alttree:</code> added check for <code>glsnogroupskip</code> 267	<code>\gls@Alphpage:new</code> 152
<code>index:</code> added check for <code>glsnogroupskip</code> 261	<code>\gls@alphpage:new</code> 152
<code>nogroupskip:new</code> 9	<code>\gls@disablepagerefexpansion:</code> new 152
<code>long:</code> added check for <code>glsnogroupskip</code> 234	<code>\gls@numberpage:new</code> 152
<code>long3col:</code> added check for <code>glsnogroupskip</code> 235	<code>\gls@protected@pagefmts:</code> new 152
<code>long4col:</code> added check for <code>glsnogroupskip</code> 236	<code>\gls@romanpage:new</code> 152
<code>longragged:</code> added check for <code>glsnogroupskip</code> 240	<code>\glsdefmain:</code> added check for doc package 12
<code>longragged3col:</code> added check for <code>glsnogroupskip</code> 241	<code>\glsorg@endtheglossary:new</code> . 5
<code>nopostdot:new</code> 9	<code>\glsorg@glossary:new</code> 4
<code>\printglossary:</code> allow title to override default <code>toctitle</code> 157	<code>\glsorg@theglossary:new</code> 5
<code>tree:</code> added check for <code>glsnogroupskip</code> 263	<code>\glsorg@wrglossary:new</code> 4
<code>treename:</code> added check for <code>glsnogroupskip</code> 264	<code>altlist:</code> replaced <code>\newline</code> with paragraph break 231
<code>super:</code> added check for <code>glsnogroupskip</code> 249	<code>\PrintChanges:new</code> 5
<code>super3col:</code> added check for <code>glsnogroupskip</code> 251	<code>\printglossary:</code> Moved aux write to end of document to prevent unwanted <code>whatsit</code> oc- curring here. 159
<code>super4col:</code> added check for <code>glsnogroupskip</code> 252	
<code>superragged:</code> added check for <code>glsnogroupskip</code> 256	
<code>superragged3col:</code> added check for <code>glsnogroupskip</code> 257	
3.04	3.05
<code>\@do@wrglossary:</code> changed <code>\theglstrycounter</code> back to <code>\@glslocref</code> 154	<code>\@do@wrglossary:</code> add Roman case. Fixed bugs in the <code>else</code> statements 153
modified to compensate for possible incorrect page num- ber 152	<code>\@gls@link:</code> added check for “no- hypertypes” 88
<code>\@gls@escbsdq:</code> unsani- tize <code>\gls@numberpage</code> , <code>\gls@alphpage</code> , <code>\gls@Alphpage</code> and <code>\gls@romanpage</code> 91	<code>\@gls@nohyperlist:new</code> 15
	<code>mcolalttree:</code> replaced ‘2’ with <code>\glsmcols</code> 247
	<code>mcolindex:</code> replaced ‘2’ with <code>\glsmcols</code> 245
	<code>mcoltree:</code> replaced ‘2’ with <code>\glsmcols</code> 245
	<code>mcoltreename:</code> replaced ‘2’ with <code>\glsmcols</code> 246
	<code>\gls@protected@pagefmts:</code> added Roman to list 152
	<code>\gls@Romanpage:new</code> 152
	<code>\GlsDeclareNoHyperList:new</code> 15
	<code>\glsgetgrouplabel:</code> fixed bug (typo in <code>\equal</code>) 169
	<code>\nopostdesc:</code> made robust 29
	<code>nohypertypes:new</code> 16

3.06		\gls@assign@descplural@field:	
	\@xdy@main@language: Changed	new	17
	back to using \language name	\gls@assign@field: new	59
	\findrootlanguage: Obsoleted	\gls@ifnotmeasuring: new	73
3.07		\glsaddallunused: new	140
	\@gls@link: fixed bug that failed	\glsexpandfields: new	59
	to find entry in list	\glsnoexpandfields: new	59
	\glossarypreamble: modified to	\glssee: made robust	155
	work with \setglossarypreamble	\glsseeformat: made robust	155
	\glsseeitem: made robust	156
	\gls@doclearpage: added check	\glsseelist: made robust	155
	for openright	\ifglsgdescsuppressed: new	48
	\glspostdescription: Added	\ifglshasdesc: new	48
	spacefactor code	\ifglshassymbol: new	48
	\GlsSetXdyCodePage: Added	list: updated list style to	
	check for fontspec	use \glossentry and	
	\SetDescriptionAcronymDisplayStyle:	\subglossentry	230
	now using \glsdoparenifnotempty	listdotted: updated listdotted	
	style to use \glossentry and	
	\setglossarypreamble: new	\subglossentry	232
3.08a		altlist: updated altlist style	
	\@glo@storeentry: no longer	to use \glossentry and	
	need to check for special char-	\subglossentry	231
	acters in any of the fields other	altlongragged4col: updated	
	than sort	to use \glossentry and	
	updated for \glossentry	\subglossentry	242
	\@glossaryentryfield: switched	alttree: updated to use	
	to \glossentry	\glossentry and \subglossentry	
	\@glossarysubentryfield:	265
	switched to \subglossentry	index: added paragraph break at	
	General: added nogroupskip key	end of environment	261
	to \printglossary	updated to use \glossentry	
	removed definition of	and \subglossentry	261
	\@glossaryentryfield	inline: updated inline style	
	..	to use \glossentry and	
	removed definition of	\subglossentry	228
	\@glossarysubentryfield	long: updated to use \glossentry	
	308	and \subglossentry	233
	\compatibleglossentry: new	longragged: updated to	
	165	use \glossentry and	
	\compatiblesubglossentry:	\subglossentry	239
	new	longragged3col: updated	
	to use \glossentry and	
	\glossaryentryfield: depre-	\subglossentry	241
	cated	tree: updated to use \glossentry	
	and \subglossentry	262
	\Glossentrydesc: new	\setglossarystyle: new	169
	\setglossentrycompatibility:	
	\glossentrydesc: new	new	166
	165		
	\Glossentryname: new		
		
	\glossentryname: new		
		
	\Glossentrysymbol: new		
		
	\glossentrysymbol: new		
		
	\gls@assign@desc@field: new		
	17		

superragged: updated to	new	66
use \glossentry and	\glswritedefhook: new	65
\subglossentry	\makeglossaries: Added	
3.09a	providecommand code to aux	
\@gls@assign@symbolplural@field:	file	148
new	\new@glossaryentry: new	60
\@gls@default@value: new ...	\newglossary: added \@gls@provide@newglossary	
\Glsentrydesc: made robust	51
\Glsentrydescplural: made ro-	\printglossary: Added provide-	
bust	command code to aux file ..	159
\Glsentryfirst: made robust .	3.11a	
\Glsentryfirstplural: made	\@ACRlong: added \glslabel,	
robust	\glsifplural, \glscapscase,	
\Glsentryfull: made robust ..	\glsinsert and \glscustomtext	307
\Glsentryfullpl: made robust	\@ACRshort: added \glslabel,	
\Glsentrylong: made robust ..	\glsifplural, \glscapscase,	
\Glsentrylongpl: made robust	\glsinsert and \glscustomtext	306
\Glsentryname: made robust ..	\@Acrlong: added \glslabel,	
\Glsentryplural: made robust	\glsifplural, \glscapscase,	
\Glsentryshort: made robust .	\glsinsert and \glscustomtext	307
\Glsentryshortpl: made robust	307
.....	\@Acrshort: added \glslabel,	
\Glsentrysymbol: made robust	\glsifplural, \glscapscase,	
\Glsentrysymbolplural: made	\glsinsert and \glscustomtext	306
robust	306
\Glsentrytext: made robust ..	\@GLS@: add \glslabel,	
\Glsentryuseri: made robust .	\glsifplural, \glscapscase,	
\Glsentryuserii: made robust	\glscustomtext and	
\Glsentryuseriii: made robust	\glsinsert	102
.....	change to using \glsentryfmt	
\Glsentryuseriv: made robust	style commands	102
\Glsentryuserv: made robust .	removed \MakeUppercase	
\Glsentryuservi: made robust	(now moved to \glsentryfmt)	
\glstextup: new	102
\if@gls@docloaded: Add a fix for	\@GLSpl: add \glslabel,	
\RecordChanges	\glsifplural, \glscapscase,	
4	\glscustomtext and	
\ifglshassymbol: changed test	\glsinsert	105
to check for \@gls@default@symbol	change to using \glsentryfmt	
.....	style commands	105
3.10a	removed \MakeUppercase	
\@gls@keymap: new	as now dealt with in	
\@gls@provide@newglossary:	\glsentryfmt	105
new	\@Gls@: add \glsifplural,	
\@gls@defaultplural: Obsolete .	\glscapscase, \glscustomtext	
\@gls@nodesc: new	and \glsinsert	100
\gls@assign@type@field: new		
17		
\gls@defglossaryentry:		
Changed to using \@gls@default@value		
.....		
66		

change to using \glentryfmt style commands	101	changed to just use \glentrydesc	113, 114
removed \makefirstuc (now dealt with in \glentryfmt)	101	changed to just use \Glsentryfirstplural	111
\@Glspl@: add \glsifplural, \glscapscase, \glscustomtext and \glsinsert	104	changed to just use \glentryfirstplural	111, 112
change to using \glentryfmt style commands	104	changed to just use \Glsentryfirst	109
removed \makefirstuc (now dealt with in \glentryfmt)	104	changed to just use \glentryfirst	109
\@acrlong: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext	307	changed to just use \Glsentryname	112
\@acrshort: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext	305	changed to just use \glentryname	112, 113
\@gls@: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert	99	changed to just use \Glsentryplural	110
change to using \glentryfmt style commands	99	changed to just use \glentryplural	110
\@gls@noexpand@fields: Fixed bug expand replaced with noexpand	58	changed to just use \Glsentrysymbolplural	117
\@glsdisp: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert	106	changed to just use \glentrysymbolplural	116, 117
change to using \glentryfmt style commands	107	changed to just use \Glsentrysymbol	115
\@Glspl@: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert	103	changed to just use \glentrysymbol	115, 116
General: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext	124–132	Changed to just use \Glsentrytext	108
changed to just use \Glsentrydescplural	114	changed to just use \glentrytext	108
changed to just use \glentrydescplural	114, 115	changed to just use \Glsentryuseriii	120
changed to just use \Glsentrydesc	113	changed to just use \glentryuseriii	119, 120
		changed to just use \Glsentryuserii	119
		changed to just use \glentryuserii	118, 119
		changed to just use \Glsentryuseriv	121
		changed to just use \glentryuseriv	120, 121
		changed to just use \Glsentryuseri	118
		changed to just use \glentryuseri	117, 118
		changed to just use \Glsentryuservi	123

changed to just use \glentryuservi	208
.....	123
changed to just use \Glsentryuserv	122
.....	122
changed to just use \glentryuserv	122
.....	122
Now requires textcase	3
acronymlists: replaced	
\@addtoacronymlists with	
\DeclareAcronymList	15
\defglstdisplay: obsoleted	85
\defglstdisplayfirst: obso-	
leted	85
\defglsentryfmt: new	50
\forglsentries: replaced \ifx	
with \ifdefempty	45
\gls@assign@desc: new	65
\gls@defglossaryentry: Fixed	
default counter if none sup-	
plied	69
\gls@doentryfmt: new	50
\glsdisplay: obsoleted	85
\glsdisplayfirst: obsoleted	85
\glsgenentryfmt: new	80
\glsgetgrouptitle: Added	
check in case non-Latin alpha-	
bet in use	168
\glsglossarymark: replaced	
\MakeUppercase with	
\mfirstucMakeUppercase	33
\glsnavigation: switched to us-	
ing \@gls@getgrouptitle	227
\ifglshasdesc: replaced	
\ifdefempty with \ifcsemt	
y	48
.....	48
\ifglshaslong: new	48
\ifglshasshort: new	48
\ifglshassymbol: replaced	
\ifdefempty with \ifcsemt	
y	48
.....	48
\ifglused: replaced \ifthenelse	
with \ifbool	46
\longnewglossaryentry: new	65
\newglossary: replaced	
\glsdisplay and \glsdisplayfirst	
with \glentryfmt	51
compatible-3.07: cnew	22
\SetCustomDisplayStyle: up-	
dated to use \defglsentryfmt	
.....	192
\SetDefaultAcronymDisplayStyle:	
changed to use \defglsentryfmt	
.....	192
\SetDescriptionAcronymDisplayStyle:	
updated to use \defglsentryfmt	
.....	198
\SetDescriptionDUAAcronymDisplayStyle:	
updated to use \defglsentryfmt	
.....	197
\SetDescriptionFootnoteAcronymDisplayStyle:	
updated to use \defglsentryfmt	
.....	194
\SetDUADisplayStyle: updated	
to use \defglsentryfmt	206
\SetFootnoteAcronymDisplayStyle:	
updated to use \defglsentryfmt	
.....	201
\SetSmallAcronymDisplayStyle:	
updated to use \defglsentryfmt	
.....	203
\setupglossaries: new	24
\showglolong: new	214
\showgloshort: new	214
numbers: new	23
symbols: new	23
3.12a	
\gls@defglossaryentry: added	
\glslabel	66
\glsaddkey: new	62
3.13a	
\@gls@assign@symbol@field:	
changed to use \glsetnoexpandfield	
.....	17
\@gls@assign@symbolplural@field:	
changed to use \glsetnoexpandfield	
.....	17
\@gls@link: removed \relax	88
\@gls@notranslatorhook: new	19
\@gls@setupsort@standard:	
moved \@gls@santizesort	
to \glsprestandardsort	10
General: added cs@gls@notranslatorhook	
to else clause	29
ucmark: added check for memoir	9
see: added \gls@checkseeallowed	
.....	55
\glossarysection: changed	
\glossarymark to \glsglossarymark	

..... 33	longheader: switched to
\glossarystyle: fixed bug	\tabularnewline 234
caused by using \ifdef in-	longheaderborder: switched to
stead of \ifcsdef 170	\tabularnewline 234
\gls@assign@desc@field:	\SetFootnoteAcronymDisplayStyle:
changed to use \glsetnoexpandfield	fixed missing argument bug 201
..... 17	super: switched to \tabularnewline
\gls@assign@descplural@field: 249
changed to use \glsetnoexpandfieldsuper3col:	switched to
..... 17	\tabularnewline 250
\gls@assign@name@field:	super3colheader: switched to
changed to use \glsetnoexpandfield	\tabularnewline 251
..... 17	super4col: switched to
\gls@assign@type@field:	\tabularnewline 252
changed to use \glsetexpandfield	super4colheader: switched to
..... 17	\tabularnewline 252
\gls@checkseeallowed: new .. 55	super4colheaderborder:
\glsaddallunused: set default to	switched to \tabularnewline
\@glo@types 140 253
\Glsentryfull: changed to use	superheader: switched to
\acrfullformat 137	\tabularnewline 249
\glsentryfull: changed to use	superheaderborder: switched to
\acrfullformat 137	\tabularnewline 250
\Glsentryfullpl: changed to 3.14a	
use \acrfullformat 137	\@glswritefiles: renamed
\glsentryfullpl: changed to	\glswritefiles to \@glswritefiles
use \acrfullformat 137	and used "savewrites" option
\glsglossarymark: renamed	to set \glswritefiles 150
\glossarymark to \glsglossarymark	General: new 216
to avoid conflict with memoir 33	acronyms: new 13
\glsprestandardsort: new ... 10	\gls@defglossaryentry: added
\glsetnoexpandfield: new .. 16	check for existence of default
altsuper4colheader: switched	glossary 67
to \tabularnewline 254	set the default for firstplural to
altsuper4colheaderborder:	be the value of plural 69
switched to \tabularnewline	xindygloss: new 22
..... 254	\longprovideglossaryentry:
long: switched to \tabularnewline	new 65
..... 233, 234	compatible-2.07: added check
long3col: switched to \tabularnewline	for 2.07 before setting 3.07
..... 235	compatibility 23
long3colheader: switched to	notranslate: new 19
\tabularnewline 235	\provideglossaryentry: new . 60
long3colheaderborder: switched 4.0	
to \tabularnewline 236	\gls@defglossaryentry: added
long4col: switched to \tabularnewline	check for first key 69
..... 236 4.01	
long4colheader: switched to	General: fixed non-value options
\tabularnewline 237	so that they can be passed to

document class	7	long-sc-short-desc: new ...	184
\CustomAcronymFields: in-		long-short: new	182
serted missing comma	209	long-short-desc: new	184
4.02		long-sm-short: new	183
\@acrfull: now using \acrfullfmt		long-sm-short-desc: new ...	185
.....	175	footnote: new	188
\@gls@indexdef: new	23	footnote-desc: new	190
\@gls@numbersdef: new	23	footnote-sc: new	189
\@gls@symbolsdef: new	23	footnote-sc-desc: new	190
General: Removed \acronymfont		footnote-sm: new	189
.....	128–132	footnote-sm-desc: new	190
\ACRfullfmt: new	177	\setacronymstyle: new	181
\Acrfullfmt: new	176	\SetDescriptionAcronymDisplayStyle:	
\acrfullfmt: new	176	Moved check for empty cus-	
\ACRfullplfmt: new	178	tom text to prevent unwanted	
\Acrfullplfmt: new	178	parenthetical material	198
\acrfullplfmt: new	177	\SetDescriptionFootnoteAcronymDisplayStyle:	
\acronymentry: new	180	Moved check for empty cus-	
sanitize: fixed bug that caused		tom text to prevent unwanted	
an error here	18	parenthetical material	194
sc-short-long: new	184	\SetFootnoteAcronymDisplayStyle:	
sc-short-long-desc: new ...	185	Moved check for empty cus-	
\Genacrfullformat: new	84	tom text to prevent unwanted	
\genacrfullformat: new	84	parenthetical material	201
\GenericAcronymFields: new	180	\SetGenericNewAcronym: new	179
\Genplacrfullformat: new ...	85	\SetSmallAcronymDisplayStyle:	
\genplacrfullformat: new ...	84	Moved check for empty cus-	
\Glsentryfull: bug fix: added		tom text to prevent unwanted	
missing \acronymfont	137	parenthetical material	203
\glentryfull: bug fix: added		dua: new	186
missing \acronymfont	137	dua-desc: new	188
\Glsentryfullpl: bug fix: added		numberedsection: added	
missing \acronymfont	137	nameref option	6
\glentryfullpl: bug fix: added		4.03	
missing \acronymfont	137	\@do@wrglossary: added	
\glsgenacfmt: new	82	\glsdetoklabel	153
\GlsUseAcrEntryDisplayStyle:		\@ACRlong: removed \glslabel	
new	182	(defined in \@gls@link) ...	307
\GlsUseAcrStyleDefs: new ..	182	\@ACRshort: removed \glslabel	
short-long: new	182	(defined in \@gls@link) ...	306
short-long-desc: new	185	\@Acrlong: removed \glslabel	
xindynoglsnumbers: new	22	(defined in \@gls@link) ...	307
sm-short-long: new	184	\@Acrshort: removed \glslabel	
sm-short-long-desc: new ...	185	(defined in \@gls@link) ...	306
\makeglossaries: made pream-		\@GLS@: removed \glslabel (de-	
ble only	149	finied in \@gls@link)	102
index: new	23	\@GLSpl: removed \glslabel	
\newacronymstyle: new	181	(defined in \@gls@link) ...	105
long-sc-short: new	183		

\@Gls@: removed \glslabel (defined in \@gls@link)	100	\gls@defglossaryentry: added	
\@Gls@entry@field: new	132	\glsdetoklabel	66
\@Glspl@: removed \glslabel (defined in \@gls@link)	104	replaced #1 with \@gls@label	67
\@acrlong: removed \glslabel (defined in \@gls@link)	307	replaced \ifthenelse with \ifdefequal	67
\@acrshort: removed \glslabel (defined in \@gls@link)	305	\glsadd: added \glsdetoklabel	139
\@Gls@: removed \glslabel (defined in \@gls@link)	99	\glsaddkey: switched to using \@gls@field@link	63
\@Gls@access@display: new	294	\glsdetoklabel: new	46
\@Gls@entry@field: new	132	\glsdisplaynumberlist: added \glsdetoklabel	138
\@Gls@fetchfield: new	62	\glsdoifexistsorwarn: new	47
\@Gls@field@link: new	107	\glsentryaccess: switched to using \@gls@entry@field	292
\@Gls@link: added \glsdetoklabel	88	\glsentrydescaccess: switched to using \@gls@entry@field	293
moved \@gls@link@opts and \@gls@link@label to \@gls@link	88	\glsentrydescpluralaccess: switched to using \@gls@entry@field	293
\@Glsdisp: removed \glslabel (defined in \@gls@link)	106	\glsentryfirstaccess: switched to using \@gls@entry@field	292
\@Glspl@: removed \glslabel (defined in \@gls@link)	103	\glsentryfirstplural: added \glsdetoklabel	134
General: changed default to \@empty instead of \relax	22	\glsentrylongaccess: switched to using \@gls@entry@field	293
removed \glslabel (defined in \@gls@link)	124	\glsentrylongpluralaccess: switched to using \@gls@entry@field	294
sc-short-long-desc: redefined to use accessibility information	312	\glsentrypluralaccess: switched to using \@gls@entry@field	293
\compatibleglossentry: added \glsdetoklabel	288	\glsentryshortaccess: switched to using \@gls@entry@field	293
\compatiblesubglossentry: added \glsdetoklabel	288	\glsentryshortpluralaccess: switched to using \@gls@entry@field	293
\Genacrfullformat: redefined to use accessibility information	305	\glsentrysymbolaccess: switched to using \@gls@entry@field	293
\genacrfullformat: redefined to use accessibility information	305	\glsentrysymbolpluralaccess: switched to using \@gls@entry@field	293
\Genplacrfullformat: redefined to use accessibility information	305	\glsentrytextaccess: switched to using \@gls@entry@field	292
\genplacrfullformat: redefined to use accessibility information	305	\glsgenacfmt: redefined to use accessibility information	302
\glossentryname: added \glsdetoklabel	165	\glsgenentryfmt: redefined to use accessibility information	300

<code>\gls hyperlink: added \glsdetoklabel</code>	<code>\ifglsused: added \glsdetoklabel</code>
..... 139 46
<code>\glslocalreset: added</code>	<code>sm-short-long-desc: redefined</code>
<code>\glsdetoklabel</code> 74	to use accessibility informa-
<code>\glslocalunset: added</code>	tion 312
<code>\glsdetoklabel</code> 74	<code>long-sc-short-desc: redefined</code>
<code>\glsmoveentry: added \glsdetoklabel</code>	to use accessibility informa-
..... 71	tion 311
replaced <code>\ifthenelse</code> with	<code>long-short: redefined to use ac-</code>
<code>\ifdefequal</code> 71	cessibility information 309
<code>\glsrefentry: added \glsdetoklabel</code>	<code>long-short-desc: redefined to</code>
..... 163	use accessibility information 310
<code>\glsreset: added \glsdetoklabel</code>	<code>long-sm-short-desc: redefined</code>
..... 73	to use accessibility informa-
<code>\glsseelist: added \expandafter</code>	tion 311
commands 156	<code>footnote: redefined to use acces-</code>
<code>\glsstepentry: added \glsdetoklabel</code>	sibility information 315
..... 163	<code>footnote-desc: redefined to use</code>
<code>\glsstepsubentry: added</code>	accessibility information ... 317
<code>\glsdetoklabel</code> 163	<code>footnote-sc: redefined to use ac-</code>
<code>\glsunset: added \glsdetoklabel</code>	cessibility information 317
..... 74	<code>footnote-sc-desc: redefined to</code>
<code>short-long: commented spuri-</code>	use accessibility information 318
<code>ous EOL</code> 183	<code>footnote-sm: redefined to use ac-</code>
redefined to use accessibility in-	cessibility information 317
formation 310	<code>footnote-sm-desc: redefined to</code>
<code>short-long-desc: redefined to</code>	use accessibility information 318
use accessibility information 311	<code>\printglossary: added</code>
<code>\ifglsdescsuppressed: added</code>	<code>\glsdetoklabel</code> 158
<code>\glsdetoklabel</code> 48	<code>\renewacronymstyle: new</code> ... 181
fixed typo 48	<code>\showglocounter: added</code>
<code>\ifglsentryexists: added</code>	<code>\glsdetoklabel</code> 211
<code>\glsdetoklabel</code> 46	<code>\showglodesc: added \glsdetoklabel</code>
<code>\ifgls haschildren: added</code> 213
<code>\glsdetoklabel</code> 47	<code>\showglodescaccess: added</code>
<code>\ifgls hasdesc: added \glsdetoklabel</code>	<code>\glsdetoklabel</code> 324
..... 48	<code>\showglodescplural: added</code>
<code>\ifgls hasfield: new</code> 49	<code>\glsdetoklabel</code> 213
<code>\ifgls haslong: added \glsdetoklabel</code>	<code>\showglodescpluralaccess:</code>
..... 48	added <code>\glsdetoklabel</code> ... 324
<code>\ifgls hasparent: added</code>	<code>\showglofirst: added \glsdetoklabel</code>
<code>\glsdetoklabel</code> 48 211
<code>\ifgls hasshort: added</code>	<code>\showglofirstaccess: added</code>
<code>\glsdetoklabel</code> 48	<code>\glsdetoklabel</code> 324
<code>\ifgls hassymbol: added</code>	<code>\showglofirstpl: added</code>
<code>\glsdetoklabel</code> 48	<code>\glsdetoklabel</code> 211
replaced <code>\ifcempty</code> with	<code>\showglofirstpluralaccess:</code>
<code>\ifdefempty</code> and replaced	added <code>\glsdetoklabel</code> ... 324
<code>\ifx</code> with <code>\ifdefequal</code> 48	

\showgloflag:added\glsdetoklabel 214	\showglosymbolaccess: added \glsdetoklabel 324
\showgloindex:added\glsdetoklabel 214	\showglosymbolplural: added \glsdetoklabel 213
\showglolevel:added\glsdetoklabel 210	\showglosymbolpluralaccess: added\glsdetoklabel ... 324
\showglolong:added\glsdetoklabel 214	\showglotext:added\glsdetoklabel 211
\showglolongaccess: added \glsdetoklabel 324	\showglotextaccess: added \glsdetoklabel 323
\showglolongpluralaccess: added\glsdetoklabel ... 325	\showglotype:added\glsdetoklabel 211
\showgloname:added\glsdetoklabel 213	\showglouseri:added\glsdetoklabel 212
\showglonameaccess: added \glsdetoklabel 323	\showglouserii: added \glsdetoklabel 212
\showgloparent: added \glsdetoklabel 210	\showglouseriii: added \glsdetoklabel 212
\showgloplural: added \glsdetoklabel 211	\showglouseriv: added \glsdetoklabel 212
\showglopluralaccess: added \glsdetoklabel 324	\showglouserv:added\glsdetoklabel 212
\showgloshort:added\glsdetoklabel 214	\showglouservi: added \glsdetoklabel 212
\showgloshortaccess: added \glsdetoklabel 324	dua:fixed bug in \acrfullfmt . 187
\showgloshortpluralaccess: added\glsdetoklabel ... 324	fixed bug in \Acrfullplfmt . 187
\showglosort:added\glsdetoklabel 213	fixed bug in \acrfullplfmt . 187
\showglosymbol: added \glsdetoklabel 213	redefined to use accessibility in- formation 312
	dua-desc: commented spurious EOL 188
	redefined to use accessibility in- formation 315

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	
\@@do@wrglossary	152
\@@glossarysec	5
\@@glossaryseclabel	6
\@@glossarysecstar	6
\@@gls@default@entryfmt	296
\@@gls@expand@field	58
\@ACRlong	307
\@ACRshort	306
\@Acrlong	307
\@Acrshort	306
\@GLS@	101
\@GLSpl	105
\@Gls@	100
\@Gls@entry@field	132
\@Glspl@	104
\@PGLS	221
\@PGLS@	221
\@PGLSpl	222
\@PGLSpl@	222
\@Pgls	220
\@Pgls@	220
\@Pglspl	221
\@Pglspl@	221
\@acrfull	175
\@acrlong	306
\@acrshort	305
\@addtoacronymlists	14
\@delimN	172
\@delimR	172
\@disable@onlypremakeg	26
\@disable@premakecs	26
\@disabled@gl saddx dy counters	37
\@do@seeglossary	155
\@do@wrglossary	152, 269
\@glo@seeautonumberlist	7
\@glo@storeentry	71
\@glo@types	50
\@glossary	151
\@glossary@default@style	6
\@glossaryentryfield	71
\@glossarysection	34
\@glossarysubentryfield	71
\@gls	99
\@gls@	99
\@gls@link	87
\@gls@access@display	294
\@gls@addpredefinedattributes	38
\@gls@assign@symbol@field ...	17
\@gls@assign@symbolplural@field	17
\@gls@checkactual	96
\@gls@checkbar	95
\@gls@checkescactual	93
\@gls@checkescbar	94
\@gls@checkesclevel	94
\@gls@checkescquote	92
\@gls@checklevel	95
\@gls@checkmkidxchars	91
\@gls@checkquote	92
\@gls@codepage	44
\@gls@counterwithin	9
\@gls@declareoption	7
\@gls@default@value	54
\@gls@do@acronymsdef	13
\@gls@entry@field	132
\@gls@escbsdq	91
\@gls@expand@fields	58
\@gls@fetchfield	62
\@gls@field@link	107
\@gls@fixbraces	155
\@gls@getcounter	52
\@gls@getcounterprefix	154
\@gls@getgrouptitle	168
\@gls@hypergroup	226
\@gls@ifinlist	36
\@gls@indexdef	23
\@gls@keymap	61, 216
\@gls@link	88
\@gls@loadlist	8
\@gls@loadlong	8
\@gls@loadsuper	8
\@gls@loadtree	8
\@gls@makefirstuc	224
\@gls@missingnumberlist	57

\@gls@noaccess	291	\@glsminrange	141
\@gls@noexpand@field	58	\@glsnextpages	161
\@gls@noexpand@fields	58	\@glsnodesc	57
\@gls@nohyperlist	15	\@glsnoname	57
\@gls@notranslatorhook	19	\@glsnonextpages	161
\@gls@numbersdef	23	\@glsopenfile	148
\@gls@onlypremakeg	26	\@glsorder	21
\@gls@provide@newglossary ...	50	\@glspl@	103
\@gls@renewglossary	151	\@glstarget	98
\@gls@sanitizedesc	16	\@glswidestname	265
\@gls@sanitizename	17	\@glswritefiles	150
\@gls@sanitizesort	17	\@istfilename	30
\@gls@sanitizesymbol	17	\@makeglossary	147
\@gls@saveentrycounter	89	\@newglossary	52
\@gls@setacrstyle	20	\@newglossaryentryposthook ..	71
\@gls@setcounter	52	\@newglossaryentryprehook ...	71
\@gls@setupsort@def	11	\@no@post@desc	29
\@gls@setupsort@standard	10	\@nopostdesc	29
\@gls@setupsort@use	11	\@onlypremakeg	26
\@gls@startswithexpandonce ..	59	\@p@pglossarysection	34
\@gls@symbolsdef	23	\@pgls	219
\@gls@tmpb	92	\@pgls@	219
\@gls@toc	35	\@pglspl	219
\@gls@updatechecked	92	\@pglspl@	219
\@gls@xdy@Lclass@Alpha-page-numbers	40	\@sPGLS	221
.....	40	\@sPGLSpl	222
\@gls@xdy@Lclass@Appendix-page-numbers	40	\@sPglS	220
.....	40	\@sPglSpl	220
\@gls@xdy@Lclass@Roman-page-numbers	40	\@set@glo@numformat	90,270
.....	40	\@sgls	99,106
\@gls@xdy@Lclass@alpha-page-numbers	40	\@sgls@link	87
.....	40	\@spgls	218
\@gls@xdy@Lclass@arabic-page-numbers	40	\@spglspl	219
.....	40	\@wrglossary	151
\@gls@xdy@Lclass@arabic-section-numbers	40	\@xdy@main@language	21
.....	40	\@xdy@attributelist	36
\@gls@xdy@Lclass@roman-page-numbers	39	\@xdy@attributes	36
.....	39	\@xdy@language	43
\@gls@xdy@locationlist	39	\@xdy@lettergroups	44
\@gls@xdy@checkbackslash	97	\@xdy@locationclassorder	42
\@gls@xdy@checkquote	96	\@xdy@locref	36
\@gls@Alphacompositor	30,40	\@xdy@requiredstyles	42
\@gls@acronymlists	14	\@xdy@sortrules	42
\@gls@defaultplural	57	\@xdy@useralphabets	39
\@gls@defaultsort	57	\@xdy@userlocationdefs	41
\@gls@disp	106	\@xdy@userlocationnames	41
\@gls@firstletter	141		
\@gls@hypernumber	171		
\@gls@link	97		
		A	
		\Ac	192

\ac	192	footnote-sm-desc	190, 318
access (key)	289	long-sc-short	183
accsupp package	288	long-sc-short-desc ..	184, 311
\accsuppglossaryentryfield .	309	long-short	182, 309
\accsuppglossarysubentryfield		long-short-desc	184, 310
.....	309	long-sm-short	183
\Acf	192	long-sm-short-desc ..	185, 311
\acf	191	sc-short-long	184
\Acfp	192	sc-short-long-desc ..	185, 312
\acfp	192	short-long	182, 310
\Acl	191	short-long-desc	185, 311
\acl	191	sm-short-long	184
\Aclp	191	sm-short-long-desc ..	185, 312
\aclp	191	\acronymentry	180
\Acp	192	\acronymfont	
\acp	192	82, 179, 196, 200, 203, 206
\acrfootnote	194	acronymlists (option)	15
\ACRfull	176	\acronymname	26
\Acrfull	176	acronyms (option)	13
\acrfull	175, 176, 180, 189, 316	\acronymsort	180
\ACRfullfmt	177	\acronymtype	12, 174
\Acrfullfmt	176	\acrpluralsuffix	174
\acrfullfmt	176	\ACRshort	125
\acrfullformat	82, 176	\Acrshort	124
\ACRfullpl	178	\acrshort	123
\Acrfullpl	178	\ACRshortpl	127
\acrfullpl	177	\Acrshortpl	126
\ACRfullplfmt	178	\acrshortpl	125
\Acrfullplfmt	178	\Acs	191
\acrfullplfmt	177	\acs	191
\acrlinkfootnote	194	\Acsp	191
\acrlinkfullformat	176	\acsp	191
\ACRlong	129	\addglossarytocaptions	27
\Acrlong	128	\addto	27
\acrlong	127	align (environment)	73, 89
\ACRlongpl	131	altlist (style)	231
\Acrlongpl	130	altlistgroup (style)	231
\acrlongpl	130	altlisthypergroup (style)	231
\acrnameformat	179, 200	altlong4col (style)	237
\acrno linkfootnote	194	altlong4colborder (style)	238
acronym (option)	13	altlong4colheader (style)	238
acronym styles:		altlong4colheaderborder (style)	238
dua	186, 312	altlongragged4col (style)	242
dua-desc	188, 315	altlongragged4colborder (style)	243
footnote	188, 315	altlongragged4colheader (style)	243
footnote-desc	190, 317	altlongragged4colheaderborder	
footnote-sc	189, 317	(style)	244
footnote-sc-desc	190, 318	\altnewglossary	52
footnote-sm	189, 317	altsuper4col (style)	253

footnote-sc-desc (acrstyle) 190, 318
 footnote-sm (acrstyle) 189, 317
 footnote-sm-desc (acrstyle) 190, 318
 \FootnoteNewAcronymDef . 201, 321
 \forall glossaries 45
 \forall glossentries 45
 \forall glossentries 45

G

garamondx package 175
 \Genacrfullformat 84, 305
 \genacrfullformat 84, 305
 \GenericAcronymFields 180
 \Genplacrfullformat 85, 305
 \genplacrfullformat 84, 305
 \glolinkprefix 88
 glossareentry (counter) 163
 glossaries package 43,
 44, 141, 210, 216, 230, 268, 288
 glossaries-accsupp package . . . 71, 288
 \GlossariesWarning 16
 \GlossariesWarningNoLine . . . 16
 \glossary 51, 147, 151, 169
 glossary counters:
 glossaryentry 162
 glossarysubentry 162
 glossary keys:
 access 289
 counter 55
 description 53
 descriptionaccess 290
 descriptionplural 53
 descriptionpluralaccess . 290
 first 54
 firstaccess 289
 firstplural 54
 firstpluralaccess 289
 long 56
 longaccess 290
 longplural 57
 longpluralaccess 290
 name 53
 nonumberlist 55
 parent 55
 plural 54
 pluralaccess 289
 see 55
 short 56
 shortaccess 290

shortplural 56
 shortpluralaccess 290
 sort 53
 symbol 54
 symbolaccess 290
 symbolplural 54
 symbolpluralaccess 290
 text 54
 textaccess 289
 type 55
 user1 56
 user2 56
 user3 56
 user4 56
 user5 56
 user6 56
 glossary package 1, 173
 glossary styles:
 altlist 231, 276
 altlist 231
 altlistgroup 231, 276
 altlistgroup 231
 altlisthypergroup . . . 231, 276
 altlisthypergroup 231
 altlong4col . . 237, 238, 242, 278
 altlong4col 237
 altlong4colborder . . . 238, 278
 altlong4colborder 238
 altlong4colheader . . . 238, 278
 altlong4colheader 238
 altlong4colheaderborder .
 238, 279
 altlong4colheaderborder . 238
 altlongragged4col 242, 243, 280
 altlongragged4col 242
 altlongragged4colborder .
 243, 280
 altlongragged4colborder . 243
 altlongragged4colheader .
 243, 280
 altlongragged4colheader . 243
 altlongragged4colheaderborder
 244, 280
 altlongragged4colheaderborder
 244
 altsuper4col . 253, 254, 258, 287
 altsuper4col 253
 altsuper4colborder . . 254, 288
 altsuper4colborder 254

altsuper4colheader ..	254, 288	long4col	236, 237, 278
altsuper4colheader	254	long4col	236
altsuper4colheaderborder		long4colborder	237, 278
.....	254, 288	long4colborder	237
altsuper4colheaderborder	254	long4colheader	237, 278
altsuperragged4col	258–260, 286	long4colheader	237
altsuperragged4col	258	long4colheaderborder	237, 278
altsuperragged4colborder		long4colheaderborder	237
.....	260, 286	longborder	234, 277
altsuperragged4colborder	260	longborder	234
altsuperragged4colheader		longheader	234, 277
.....	259, 286	longheader	234
altsuperragged4colheader	259	longheaderborder	234, 277
altsuperragged4colheaderborder		longheaderborder	234
.....	260, 286	longragged	239–241
altsuperragged4colheaderborder		longragged	239
.....	260	longragged3col ...	241, 242, 279
alttree	247, 265, 267, 282	longragged3col	241
alttree	265	longragged3colborder	241, 279
alttreegroup	268, 283	longragged3colborder	241
alttreegroup	267	longragged3colheader	242, 279
alttreehypergroup ...	268, 283	longragged3colheader	242
alttreehypergroup	268	longragged3colheaderborder	
index	244, 260–262, 280	242, 280
index	260	longragged3colheaderborder	
indexgroup	261, 262, 281	242
indexgroup	261	longraggedborder	240, 279
indexhypergroup	262, 281	longraggedborder	240
indexhypergroup	262	longraggedheader	240, 279
inline	275	longraggedheader	240
inline	227	longraggedheaderborder	240, 279
list	6, 230–232, 276	longraggedheaderborder ..	240
list	230	mc colalttree	247, 284
listdotted	232, 276	mc colalttree	247
listdotted	232	mc colalttreegroup	247, 284
listgroup	230, 276	mc colalttreegroup	247
listgroup	230	mc colalttreehypergroup	247, 284
listhypergroup	230, 276	mc colalttreehypergroup ...	247
listhypergroup	230	mc colindex	245, 283
long	233, 234, 239, 277, 279	mc colindex	244
long	233	mc colindexgroup	245, 284
long3col	234, 235, 277	mc colindexgroup	245
long3col	234	mc colindexhypergroup .	245, 284
long3colborder	235, 277	mc colindexhypergroup	245
long3colborder	235	mc coltree	245, 284
long3colheader	235, 278	mc coltree	245
long3colheader	235	mc coltreegroup	284
long3colheaderborder	236, 278	mc coltreegroup	245
long3colheaderborder	236	mc coltreehypergroup ..	246, 284

mcoltreehypergroup	246	superraggedheaderborder .	
mcoltreenoname	246, 284	256, 285
mcoltreenoname	246	superraggedheaderborder .	256
mcoltreenonamegroup .	246, 284	superraggedright3colheaderborder	
mcoltreenonamegroup	246	258
mcoltreenonamehypergroup		tree	245, 262–265, 281
.....	246, 284	tree	262
mcoltreenonamehypergroup	246	treegroup	246, 263, 281
sublistdotted	277	treegroup	263
sublistdotted	232	treehypergroup	263, 281
super	248–250, 256, 286	treehypergroup	263
super	248	treenoname	246, 264, 282
super3col	250, 251, 286	treenoname	264
super3col	250	treenonamegroup	265, 282
super3colborder	251, 287	treenonamegroup	264
super3colborder	251	treenonamehypergroup	265, 282
super3colheader	251, 287	treenonamehypergroup	265
super3colheader	251	glossary-hypernav package	141
super3colheaderborder	251, 287	glossary-list package	6, 8, 229
super3colheaderborder ...	251	glossary-long package .	8, 233, 242, 248
super4col	252, 253, 287	glossary-longragged package	239
super4col	252	glossary-mcols package	244
super4colborder	253, 287	glossary-super package	
super4colborder	253	8, 233, 248, 255, 258
super4colheader	252, 287	glossary-superragged package	255
super4colheader	252	glossary-tree package	8, 260
super4colheaderborder	253, 287	glossaryentry (counter) .	9, 163, 164
super4colheaderborder ...	253	glossaryentry (counter)	162
superborder	249, 286	\glossaryentryfield .	167, 170, 171
superborder	249	\glossaryentrynumber ...	161, 162
superheader	249, 286	\glossaryentrynumbers	
superheader	249	7, 31, 157, 160
superheaderborder ...	250, 286	\glossaryheader	164, 170
superheaderborder	250	\glossarymark	33
superragged	255–257, 285	\glossaryname	26, 27
superragged	255	\glossarypostamble	32, 170
superragged3col ..	257, 258, 285	\glossarypreamble	32, 170
superragged3col	257	\glossarysection	6, 32, 51
superragged3colborder	257, 285	\glossarystyle	169, 210
superragged3colborder ...	257	glossarysubentry (counter) ...	
superragged3colheader	258, 285	9, 162–164
superragged3colheader ...	258	glossarysubentry (counter) ...	162
superragged3colheaderborder		\glossarysubentryfield	167
.....	258, 285	\glossentry	54, 165
superraggedborder ...	256, 285	\Glossentrydesc	166
superraggedborder	256	\glossentrydesc	165, 308
superraggedheader ...	256, 285	\Glossentryname	165
superraggedheader	256	\glossentryname	165, 308
		\Glossentrysymbol	166

<code>\glossentrysymbol</code>	166, 308	<code>\glsclearpage</code>	35
<code>\GLS</code>	101	<code>\glsclosebrace</code>	141
<code>\Gls</code>	100, 104, 223	<code>\glscompositor</code>	30, 40
<code>\gls</code>	4, 54, 75, 87, 99, 101, 102, 107–109, 111–122, 163, 218	<code>\glscounter</code>	15, 52
<code>\gls@Alphpage</code>	152	<code>\GlsDeclareNoHyperList</code>	15
<code>\gls@alphpage</code>	152	<code>\glsdefaulttype</code>	12
<code>\gls@assign@desc</code>	65	<code>\glsdefmain</code>	12
<code>\gls@assign@desc@field</code>	17	<code>\GLSdesc</code>	113
<code>\gls@assign@descplural@field</code>	17	<code>\Glsdesc</code>	113
<code>\gls@assign@field</code>	59	<code>\glsdesc</code>	113
<code>\gls@assign@name@field</code>	17	<code>\GLSdescplural</code>	115
<code>\gls@assign@type@field</code>	17	<code>\Glsdescplural</code>	114
<code>\gls@checkisacronymlist</code>	15	<code>\glsdescplural</code>	114
<code>\gls@checkseeallowed</code>	55	<code>\glsdescriptionaccessdisplay</code>	295
<code>\gls@codepage</code>	22	<code>\glsdescriptionpluralaccessdisplay</code>	295
<code>\gls@defglossaryentry</code>	65	<code>\glsdescwidth</code> ...	233, 239, 248, 255
<code>\gls@disablepagerefexpansion</code>	152	<code>\glsdetoklabel</code>	46
<code>\gls@doclearpage</code>	35	<code>\glsdisablehyper</code>	98
<code>\gls@doentryfmt</code>	50	<code>\glsdisp</code>	106
<code>\gls@hypergroup prerun</code>	226	<code>\glsdisplay</code>	76, 85, 99
<code>\gls@ifnotmeasuring</code>	73	<code>\glsdisplayfirst</code>	76, 85, 99
<code>\gls@level</code>	57	<code>\glsdisplaynumberlist</code>	138
<code>\gls@numberpage</code>	152	<code>\glsdoifexists</code>	46
<code>\gls@protected@pagefmts</code>	152	<code>\glsdoifexistsorwarn</code>	47
<code>\gls@Romanpage</code>	152	<code>\glsdoifnoexists</code>	47
<code>\gls@romanpage</code>	152	<code>\glsdoparenifnotempty</code>	203
<code>\gls@save@numberlist</code>	156	<code>\glsenablehyper</code>	98
<code>\gls@suffiX</code>	31	<code>\glsentryaccess</code>	292
<code>\gls@suffiXFF</code>	31	<code>\glsentrycounter</code>	89
<code>\glsaccessdisplay</code>	295	<code>\glsentrycounterlabel</code>	163
<code>\glsacccsupp</code>	294	<code>\Glsentrydesc</code>	133
<code>\glsadd</code>	75, 139, 169	<code>\glsentrydesc</code>	133
<code>\glsadd options</code>		<code>\glsentrydescaccess</code>	293
counter	139	<code>\Glsentrydescplural</code>	133
format	139, 171	<code>\glsentrydescplural</code>	133
<code>\glsaddall</code>	46, 75, 140	<code>\glsentrydescpluralaccess</code> ..	293
<code>\glsaddall options</code>		<code>\Glsentryfirst</code>	134
types	139, 140	<code>\glsentryfirst</code>	134
<code>\glsaddallunused</code>	140	<code>\glsentryfirstaccess</code>	292
<code>\glsaddkey</code>	62	<code>\Glsentryfirstplural</code>	135
<code>\GlsAddLetterGroup</code>	44	<code>\glsentryfirstplural</code>	134
<code>\GlsAddSortRule</code>	42	<code>\glsentryfirstpluralaccess</code> .	293
<code>\GlsAddXdyAlphabet</code>	39	<code>\glsentryfmt</code>	53, 54, 76
<code>\GlsAddXdyAttribute</code>	37, 268	<code>\Glsentryfull</code>	137
<code>\GlsAddXdyCounters</code>	36, 269	<code>\glsentryfull</code> ...	137, 180, 189, 316
<code>\GlsAddXdyLocation</code>	41, 269	<code>\Glsentryfullpl</code>	137
<code>\GlsAddXdyStyle</code>	43	<code>\glsentryfullpl</code>	137
<code>\glsautoprefix</code>	6	<code>\glsentryitem</code>	164

<code>\Glsentrylong</code>	137	<code>\glsexpandfields</code>	59
<code>\Glsentrylong</code>	137	<code>\GLSfirst</code>	109
<code>\Glsentrylongaccess</code>	293	<code>\Glsfirst</code>	109
<code>\Glsentrylongpl</code>	137	<code>\glsfirst</code>	108, 109
<code>\Glsentrylongpl</code>	137	<code>\glsfirstaccessdisplay</code>	294
<code>\Glsentrylongpluralaccess</code>	294	<code>\GLSfirstplural</code>	111
<code>\Glsentryname</code>	133	<code>\Glsfirstplural</code>	111
<code>\Glsentryname</code>	133, 156	<code>\glsfirstplural</code>	111
<code>\Glsentrynumberlist</code>	138	<code>\glsfirstpluralaccessdisplay</code>	295
<code>\Glsentryplural</code>	134	<code>\glsgenacfmt</code>	82, 302
<code>\Glsentryplural</code>	133	<code>\glsgenentryfmt</code>	80, 300
<code>\Glsentrypluralaccess</code>	293	<code>\glsgetgrouplabel</code>	169
<code>\Glsentryprefix</code>	218	<code>\glsgetgrouptitle</code>	141, 168
<code>\Glsentryprefix</code>	217	<code>\glsglossarymark</code>	9, 33
<code>\Glsentryprefixfirst</code>	217	<code>\glsgroupheading</code>	168, 170
<code>\Glsentryprefixfirst</code>	217	<code>\glsgroupskip</code>	167, 170, 230
<code>\Glsentryprefixfirstplural</code>	217	<code>\glshyperlink</code>	139
<code>\Glsentryprefixfirstplural</code>	217	<code>\glshypernavsep</code>	227
<code>\Glsentryprefixplural</code>	218	<code>\glshypernumber</code>	31, 171
<code>\Glsentryprefixplural</code>	217	<code>\glsIfListOfAcronyms</code>	14
<code>\Glsentryshort</code>	136	<code>\glsinlinedescformat</code>	229
<code>\Glsentryshort</code>	136	<code>\glsinlinedopostchild</code>	228, 229
<code>\Glsentryshortaccess</code>	293	<code>\glsinlineemptydescformat</code>	229
<code>\Glsentryshorttpl</code>	137	<code>\glsinlinenameformat</code>	229
<code>\Glsentryshorttpl</code>	137	<code>\glsinlineparentchildseparator</code>	229
<code>\Glsentryshortpluralaccess</code>	293		
<code>\Glsentrysort</code>	135	<code>\glsinlinepostchild</code>	229
<code>\Glsentrysymbol</code>	134	<code>\glsinlineseparator</code>	229
<code>\Glsentrysymbol</code>	134	<code>\glsinlinesubdescformat</code>	229
<code>\Glsentrysymbolaccess</code>	293	<code>\glsinlinesubnameformat</code>	229
<code>\Glsentrysymbolplural</code>	134	<code>\glsinlinesubseparator</code>	229
<code>\Glsentrysymbolplural</code>	134	<code>\glskeylisttok</code>	179
<code>\Glsentrysymbolpluralaccess</code>	293	<code>\glslabeltok</code>	179
<code>\Glsentrytext</code>	133	<code>\glslink</code>	75, 76, 87, 99, 139, 169, 171
<code>\Glsentrytext</code>	46, 133, 156	<code>\glslink options</code>	
<code>\Glsentrytextaccess</code>	292	counter	86, 99, 216
<code>\Glsentrytype</code>	135	format	86, 99, 171
<code>\Glsentryuseri</code>	135	hyper	87, 99
<code>\Glsentryuseri</code>	135	local	87
<code>\Glsentryuserii</code>	135	<code>\glslistdottedwidth</code>	232
<code>\Glsentryuserii</code>	135	<code>\glslocalreset</code>	74
<code>\Glsentryuseriii</code>	136	<code>\glslocalresetall</code>	74
<code>\Glsentryuseriii</code>	135	<code>\glslocalunset</code>	74
<code>\Glsentryuseriv</code>	136	<code>\glslocalunsetall</code>	75
<code>\Glsentryuseriv</code>	136	<code>\gslongaccessdisplay</code>	295
<code>\Glsentryuserv</code>	136	<code>\gslongaccesskey</code>	323
<code>\Glsentryuserv</code>	136	<code>\gslongkey</code>	175
<code>\Glsentryuservi</code>	136	<code>\gslongpluralaccessdisplay</code>	295
<code>\Glsentryuservi</code>	136	<code>\gslongpluralaccesskey</code>	323

<code>\glslongpluralkey</code>	175	<code>\glsseeitemformat</code>	156
<code>\glslongtok</code>	179	<code>\glsseelastsep</code>	156
<code>\glsmakefirstuc</code>	224	<code>\glsseelist</code>	155
<code>\glsmcols</code>	244	<code>\glsseesep</code>	156
<code>\glsmoveentry</code>	71	<code>\glsSetAlphaCompositor</code>	30
<code>\GLSname</code>	112	<code>\glsSetCompositor</code>	30
<code>\Glsname</code>	112	<code>\glssetnoexpandfield</code>	16
<code>\glsname</code>	112	<code>\glsSetSuffixF</code>	31
<code>\glsnameaccessdisplay</code>	294	<code>\glsSetSuffixFF</code>	31
<code>\glsnamefont</code>	171	<code>\glssettoctitle</code>	26
<code>\glsnavhyperlink</code>	225	<code>\glssetwidest</code>	265
<code>\glsnavhypertarget</code>	225	<code>\GlsSetXdyCodePage</code>	44
<code>\glsnavigation</code>	226	<code>\GlsSetXdyFirstLetterAfterDigits</code>	141
<code>\glsnextpages</code>	162	<code>\GlsSetXdyLanguage</code>	43
<code>\glsnoexpandfields</code>	59	<code>\GlsSetXdyLocationClassOrder</code>	42
<code>\glsnoextpages</code>	162	<code>\GlsSetXdyMinRangeLength</code>	141
<code>\glsnoxindywarning</code>	36	<code>\GlsSetXdyStyles</code>	43
<code>\glsnumberformat</code>	31	<code>\glsshortaccessdisplay</code>	295
<code>\glsnumbersgroupname</code>	27, 141, 168	<code>\glsshortaccesskey</code>	323
<code>\glsnumlistlastsep</code>	139	<code>\glsshortkey</code>	175
<code>\glsnumlistsep</code>	139	<code>\glsshortpluralaccessdisplay</code>	295
<code>\glsopenbrace</code>	141	<code>\glsshortpluralaccesskey</code>	323
<code>\glsorder</code>	21	<code>\glsshortpluralkey</code>	175
<code>\glsorg@endtheglossary</code>	5	<code>\glsshorttok</code>	179
<code>\glsorg@glossary</code>	4	<code>\glssortnumberfmt</code>	10
<code>\glsorg@theglossary</code>	5	<code>\glsstepentry</code>	163
<code>\glsorg@wrglossary</code>	4	<code>\glsstepsubentry</code>	163
<code>\glspagelistwidth</code>	233, 239, 248, 255	<code>\glssubentrycounterlabel</code>	164
<code>\glspar</code>	29	<code>\glssubentryitem</code>	164
<code>\GLSpl</code>	105	<code>\GLSsymbol</code>	116
<code>\Glspl</code>	104, 223	<code>\Glssymbol</code>	115
<code>\glspl</code>	75, 102, 104, 105	<code>\glssymbol</code>	115, 116
<code>\GLSplural</code>	110	<code>\glssymbolaccessdisplay</code>	295
<code>\Glsplural</code>	110	<code>\glssymbolnav</code>	227
<code>\glsplural</code>	110	<code>\GLSsymbolplural</code>	117
<code>\glspluralaccessdisplay</code>	294	<code>\Glssymbolplural</code>	116
<code>\glspluralsuffix</code>	27, 54	<code>\glssymbolplural</code>	116, 117
<code>\glspostdescription</code>	8	<code>\glssymbolpluralaccessdisplay</code>	295
<code>\glspostinline</code>	229	<code>\glssymbolsgroupname</code>	27, 141, 168
<code>\glsprestandardsort</code>	10	<code>\glstarget</code>	164
<code>\glsquote</code>	141	<code>\GLStext</code>	108
<code>\glsrefentry</code>	163	<code>\Glstext</code>	108
<code>\glsreset</code>	73	<code>\glstext</code>	107
<code>\glsresetall</code>	74	<code>\glstextaccessdisplay</code>	294
<code>\glsresetentrylist</code>	162	<code>\glstextformat</code>	76
<code>\glsresetsubentrycounter</code>	162, 163	<code>\glstextup</code>	175
<code>\glssee</code>	155	<code>\glstreeindent</code>	263, 264
<code>\glsseeformat</code>	143, 155		
<code>\glsseeitem</code>	156		

<code>\glsunset</code>	74
<code>\glsunsetall</code>	75
<code>\GlsUseAcrEntryDispStyle</code> ...	182
<code>\GlsUseAcrStyleDefs</code>	182
<code>\GLSuseri</code>	118
<code>\Glsuseri</code>	117
<code>\glsuseri</code>	117, 118
<code>\GLSuserii</code>	119
<code>\Glsuserii</code>	118
<code>\glsuserii</code>	118, 119
<code>\GLSuseriii</code>	120
<code>\Glsuseriii</code>	120
<code>\glsuseriii</code>	119, 120
<code>\GLSuseriv</code>	121
<code>\Glsuseriv</code>	121
<code>\glsuseriv</code>	120, 121
<code>\GLSuserv</code>	122
<code>\Glsuserv</code>	122
<code>\glsuserv</code>	121, 122
<code>\GLSuservi</code>	123
<code>\Glsuservi</code>	123
<code>\glsuservi</code>	122, 123
<code>\glswrite</code>	150
<code>\gls.writedefhook</code>	65

H

<code>\hyperbf</code>	173
<code>\hyperemph</code>	173
<code>hyperfirst</code> (option)	20
<code>\hyperit</code>	173
<code>\hyperlink</code>	97
<code>\hypermd</code>	173
<code>\hyperpage</code>	171
<code>hyperref</code> package ...	154, 157, 171, 216
<code>\hyperm</code>	173
<code>\hypersc</code>	173
<code>\hypersf</code>	173
<code>\hypersl</code>	173
<code>\hypertarget</code>	98
<code>\hypertt</code>	173
<code>\hyperup</code>	173

I

<code>\if@gls@docloaded</code>	4
<code>\if@gls@isacronymlist</code>	15
<code>\ifglossaryexists</code>	45
<code>\ifglsdescsuppressed</code>	48
<code>\ifglsentryexists</code>	46
<code>\ifglshaschildren</code>	47
<code>\ifglshasdesc</code>	48

<code>\ifglshasfield</code>	49
<code>\ifglshaslong</code>	48
<code>\ifglshasparent</code>	48
<code>\ifglshasprefix</code>	218
<code>\ifglshasprefixfirst</code>	218
<code>\ifglshasprefixfirstplural</code> .	218
<code>\ifglshasprefixplural</code>	218
<code>\ifglshasshort</code>	48
<code>\ifglshassymbol</code>	48
<code>\ifglstranslate</code>	19
<code>\ifglsused</code>	46, 73
<code>\ifglsexindy</code>	21
<code>index</code> (option)	23
<code>index</code> (style)	260
<code>indexgroup</code> (style)	261
<code>indexhypergroup</code> (style)	262
<code>indexonlyfirst</code> (option)	20
<code>inline</code> (style)	227
<code>\inputencodingname</code>	22
<code>\istfile</code>	150
<code>\istfilename</code>	30
<code>\item</code>	171, 229, 261

L

<code>link text</code>	76
<code>list</code> (style)	230
<code>listdotted</code> (style)	232
<code>listgroup</code> (style)	230
<code>listhypergroup</code> (style)	230
<code>\loadglsentries</code>	12, 75
<code>long</code> (key)	56
<code>long</code> (style)	233
<code>long-sc-short</code> (acrstyle)	183
<code>long-sc-short-desc</code> (acrstyle) .	184, 311
<code>long-short</code> (acrstyle)	182, 309
<code>long-short-desc</code> (acrstyle) .	184, 310
<code>long-sm-short</code> (acrstyle)	183
<code>long-sm-short-desc</code> (acrstyle) .	185, 311
<code>long3col</code> (style)	234
<code>long3colborder</code> (style)	235
<code>long3colheader</code> (style)	235
<code>long3colheaderborder</code> (style) ..	236
<code>long4col</code> (style)	236
<code>long4colborder</code> (style)	237
<code>long4colheader</code> (style)	237
<code>long4colheaderborder</code> (style) ..	237
<code>longaccess</code> (key)	290

longborder (style) 234
 longheader (style) 234
 longheaderborder (style) 234
 \longnewglossaryentry 53, 65
 longplural (key) 57
 longpluralaccess (key) 290
 \longprovideglossaryentry ... 65
 longragged (style) 239
 longragged3col (style) 241
 longragged3colborder (style) .. 241
 longragged3colheader (style) .. 242
 longragged3colheaderborder
 (style) 242
 longraggedborder (style) 240
 longraggedheader (style) 240
 longraggedheaderborder (style) 240
 longtable (environment)
 8, 210, 233–244
 longtable package 233, 239

M

\makefirstuc 223
 makeglossaries 21, 30, 43, 44, 51, 159
 \makeglossaries
 26, 29–31, 50, 52, 148, 149
 \makeglossary 149
 makeindex 338
 makeindex .. 10, 21, 27, 29–31, 51–
 53, 72, 90, 93, 140, 143, 146,
 147, 153, 154, 167, 168, 269, 270
 delim_n 31
 delim_r 31
 page_compositor 30
 special characters 91, 92, 140
 makeindex (option) 21
 mcolalttree (style) 247
 mcolalttreegroup (style) 247
 mcolalttreehypergroup (style) . 247
 mcolindex (style) 244
 mcolindexgroup (style) 245
 mcolindexhypergroup (style) ... 245
 mcoltree (style) 245
 mcoltreegroup (style) 245
 mcoltreehypergroup (style) 246
 mcoltreenoname (style) 246
 mcoltreenonamegroup (style) ... 246
 mcoltreenonamehypergroup
 (style) 246
 memoir class 151

mfirstuc package 1
 \mfirstucMakeUppercase 224
 multicol package 244
 multicol (environment) 244

N

name (key) 53
 \new@glossaryentry 60
 \newacronym
 ... 20, 21, 56, 57, 75, 76, 173, 174
 \newacronymhook 179
 \newacronymstyle 181
 \newglossary 15, 51, 52, 147, 149, 160
 \newglossaryentry 53, 59, 75, 76, 174
 \newglossaryentry options
 access 291, 292
 counter 55
 description 20, 53,
 57, 59, 66, 113, 133, 174, 202, 290
 descriptionaccess 293, 295
 descriptionplural 114, 290
 descriptionpluralaccess .. 293, 295
 first 54, 69, 99, 108, 134, 200, 205, 289
 firstaccess 292, 294
 firstplural 54, 111, 134, 289
 firstpluralaccess 293, 295
 format 142
 long 82, 137, 290
 longaccess 293, 295
 longplural 137, 290
 longpluralaccess 294, 295
 name 53,
 54, 57, 59, 66, 112, 132, 156, 289
 nonumberlist 55
 parent 55, 59
 plural 54, 69, 109, 289
 pluralaccess 293, 294
 prefix 217
 prefixfirst 217
 prefixfirstplural 217
 prefixplural 217
 see 7, 55, 149
 short 82, 136, 290
 shortaccess 293, 295
 shortplural 137, 290
 shortpluralaccess 293, 295
 sort 53, 135, 167, 168
 symbol 53, 54, 115, 196,
 198, 200, 205, 236, 252, 289–291

symbolaccess 293, 295
 symbolplural 116, 290
 symbolpluralaccess 293, 295
 text . 54, 99, 107, 133, 196, 200, 289
 textaccess 292, 294
 type 12, 55, 75, 135
 user1 117, 135, 290
 user2 118, 135
 user3 119, 135
 user4 120, 136
 user5 121, 136
 user6 122, 136, 290
 \newglossarystyle 170
 nogroupskip (option) 9
 nohypertypes (option) 16
 \noist 147, 216, 274
 nolist (option) 8
 nolong (option) 8
 nomain (option) 13
 nonumberlist (key) 55
 nonumberlist (option) 7
 \nopostdesc 29
 nopostdot (option) 9
 nostyles (option) 8
 nosuper (option) 8
 notranslate (option) 19
 notree (option) 8
 nowarn (option) 16
 numberedsection (option) 6
 numberline (option) 5
 numbers (option) 23

O

\oldacronym 173
 order (option) 21

P

package options:

acronym 12, 13, 26, 160, 174
 true 13
 acronym 13
 acronymlists 15
 acronyms 13
 compatible-2.07 22
 compatible-3.07 22
 counter 15
 counter 15
 description 200, 201
 description 20
 dua 198, 200, 201

dua 21
 entrycounter 162
 true 9
 entrycounter 9
 entrycounterwithin 9
 footnote 99,
 101–105, 107, 196, 198, 200, 202
 footnote 20
 hyperfirst
 false 99, 101–105, 107
 hyperfirst 20
 index 23
 indexonlyfirst 345
 indexonlyfirst 20
 makeindex 144, 216
 makeindex 21
 nogroupskip 9
 nohypertypes 16
 nolist 210
 nolist 8
 nolong 210, 233
 nolong 8
 nomain 12, 13
 nomain 13
 nonumberlist 7
 nonumberlist 7
 nopostdot 9
 nostyles 8
 nosuper 210
 nosuper 8
 notranslate 19
 notree 210
 notree 8
 nowarn 16
 numberedsection 6
 numberline 5
 numberline 5
 numbers 23
 order 21
 sanitize 17, 53, 132, 133
 sanitize 18
 sanitizesort 18
 savenumberlist 7
 savewrites 22, 342
 false 147
 true 150
 savewrites 22
 section 6, 34
 section 6

<code>\setglossarypreamble</code>	32	<code>\showglosymbolpluralaccess</code> .	324
<code>\setglossarysection</code>	34	<code>\showglotext</code>	211
<code>\setglossarystyle</code>	169	<code>\showglotextaccess</code>	323
<code>\setglossentrycompatibility</code>	166	<code>\showglotype</code>	211
<code>\SetSmallAcronymDisplayStyle</code>	203	<code>\showglouserii</code>	212
<code>\SetSmallAcronymStyle</code>	205	<code>\showglouseriii</code>	212
<code>\setStyleFile</code>	29	<code>\showglouseriii</code>	212
<code>\setupglossaries</code>	24	<code>\showglouseriv</code>	212
<code>short (key)</code>	56	<code>\showglouserv</code>	212
<code>short-long (acrstyle)</code>	182, 310	<code>\showglouservi</code>	212
<code>short-long-desc (acrstyle)</code> .	185, 311	<code>sm-short-long (acrstyle)</code>	184
<code>shortaccess (key)</code>	290	<code>sm-short-long-desc (acrstyle)</code> ..	
<code>shortplural (key)</code>	56	185, 312
<code>shortpluralaccess (key)</code>	290	<code>smallcaps (option)</code>	20
<code>shortcuts (option)</code>	21	<code>smaller (option)</code>	20
<code>\showacronymlists</code>	214	<code>\SmallNewAcronymDef</code>	204, 322
<code>\showglocounter</code>	211	<code>sort (key)</code>	53
<code>\showglodesc</code>	213	<code>sort (option)</code>	9
<code>\showglodescaccess</code>	324	<code>style (option)</code>	7
<code>\showglodescplural</code>	213	<code>subentrycounter (option)</code>	9
<code>\showglodescpluralaccess</code> ..	324	<code>\subglossentry</code>	165
<code>\showglofirst</code>	211	<code>\subitem</code>	261
<code>\showglofirstaccess</code>	324	<code>sublistdotted (style)</code>	232
<code>\showglofirsttpl</code>	211	<code>\subsubitem</code>	261
<code>\showglofirsttplpluralaccess</code> ..	324	<code>super (style)</code>	248
<code>\showgloflag</code>	214	<code>super3col (style)</code>	250
<code>\showgloindex</code>	214	<code>super3colborder (style)</code>	251
<code>\showglolevel</code>	210	<code>super3colheader (style)</code>	251
<code>\showglolong</code>	214	<code>super3colheaderborder (style)</code> .	251
<code>\showglolongaccess</code>	324	<code>super4col (style)</code>	252
<code>\showglolongpluralaccess</code> ..	325	<code>super4colborder (style)</code>	253
<code>\showgloname</code>	213	<code>super4colheader (style)</code>	252
<code>\showglonameaccess</code>	323	<code>super4colheaderborder (style)</code> .	253
<code>\showgloparent</code>	210	<code>superborder (style)</code>	249
<code>\showgloplural</code>	211	<code>superheader (style)</code>	249
<code>\showglopluralaccess</code>	324	<code>superheaderborder (style)</code>	250
<code>\showgloshort</code>	214	<code>superragged (style)</code>	255
<code>\showgloshortaccess</code>	324	<code>superragged3col (style)</code>	257
<code>\showgloshortpluralaccess</code> ..	324	<code>superragged3colborder (style)</code> .	257
<code>\showglosort</code>	213	<code>superragged3colheader (style)</code> .	258
<code>\showglossaries</code>	214	<code>superraggedborder (style)</code>	256
<code>\showglossarycounter</code>	215	<code>superraggedheader (style)</code>	256
<code>\showglossaryentries</code>	215	<code>superraggedheaderborder (style)</code>	256
<code>\showglossaryin</code>	215	<code>superraggedright3colheaderborder</code>	
<code>\showglossaryout</code>	215	(style)	258
<code>\showglossarytitle</code>	215	<code>supertabular (environment)</code> ...	
<code>\showglosymbol</code>	213	8, 210, 248–260
<code>\showglosymbolaccess</code>	324	<code>supertabular package</code> ..	8, 210, 248, 255
<code>\showglosymbolplural</code>	213	<code>symbol (key)</code>	54

symbolaccess (key) 290
 \symbolname 27
 symbolplural (key) 54
 symbolpluralaccess (key) 290
 symbols (option) 23

T

text (key) 54
 textaccess (key) 289
 textcase package 3
 \theequation 154
 theglossary (environment) 5, 32,
 164, 170, 246, 247, 262, 263, 265
 \theHequation 154
 theindex (environment) 261
 toc (option) 5
 \translate 27
 translate (option) 19
 translator package
 ... 25, 27, 28, 157, 325, 334–337
 tree (style) 262
 treegroup (style) 263
 treehypergroup (style) 263
 treenoname (style) 264
 treenonamegroup (style) 264
 treenonamehypergroup (style) .. 265
 type (key) 55

U

ucmark (option) 9
 user1 (key) 56
 user2 (key) 56
 user3 (key) 56
 user4 (key) 56
 user5 (key) 56
 user6 (key) 56

W

\warn@nomakeglossaries 148
 \warn@noprintglossary 157
 \writeist 29, 37, 38, 41, 141, 268, 270

X

\xcapitalisewords 225
 \xglaccsupp 294
 xindy 338
 xindy 10, 21, 22, 29, 30, 36,
 39, 41–44, 72, 96, 97, 141–143,
 153, 154, 159, 167, 216, 269, 270
 xindy (option) 22
 xindygloss (option) 22
 xindynoglsnumbers (option) 22
 \xmakefirstuc 224
 xspace package 4, 174