

Documented Code For glossaries v4.01

Nicola L.C. Talbot

Dickimaw Books

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

2013-11-16

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 v4.01: \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	25
1.4 Xindy	35
1.5 Loops and conditionals	43
1.6 Defining new glossaries	47
1.7 Defining new entries	50
1.8 Resetting and unsetting entry flags	70
1.9 Loading files containing glossary entries	72
1.10 Using glossary entries in the text	72
1.10.1 Links to glossary entries	80
1.10.2 Displaying entry details without adding information to the glossary	131
1.11 Adding an entry to the glossary without generating text	138
1.12 Creating associated files	139
1.13 Writing information to associated files	148
1.14 Glossary Entry Cross-References	153
1.15 Displaying the glossary	155
1.16 Acronyms	172
1.17 Predefined acronym styles	176
1.18 Predefined Glossary Styles	195
1.19 Debugging Commands	195
1.20 Compatibility with version 2.07 and below	200
2 Prefix Support (glossaries-prefix Code)	201
3 Mfirstuc Documented Code	207
4 Glossary Styles	209
4.1 Glossary hyper-navigation definitions (glossary-hypernav package)	209
4.2 In-line Style (glossary-inline.sty)	212
4.3 List Style (glossary-list.sty)	214
4.4 Glossary Styles using longtable (the glossary-long package)	217
4.5 Glossary Styles using longtable (the glossary-longragged package)	223
4.6 Glossary Styles using multicol (glossary-mcols.sty)	229
4.7 Glossary Styles using supertabular environment (glossary-super package)	232
4.8 Glossary Styles using supertabular environment (glossary-superragged package)	239
4.9 Tree Styles (glossary-tree.sty)	245
5 glossaries-compatible-207	253

6 Accessibility Support (glossaries-accsupp Code)	273
6.1 Defining Replacement Text	274
6.2 Accessing Replacement Text	277
6.3 Displaying the Glossary	288
6.4 Acronyms	289
6.5 Debugging Commands	293
7 Multi-Lingual Support	294
7.1 Babel Captions	294
7.2 Polyglossia Captions	300
7.3 Brazilian Dictionary	303
7.4 Danish Dictionary	304
7.5 Dutch Dictionary	304
7.6 English Dictionary	304
7.7 French Dictionary	305
7.8 German Dictionary	305
7.9 Irish Dictionary	305
7.10 Italian Dictionary	305
7.11 Magyar Dictionary	306
7.12 Polish Dictionary	306
7.13 Serbian Dictionary	306
7.14 Spanish Dictionary	307
Glossary	307
Change History	307
Index	322

1 Main Package Code

1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2013/11/16 v4.01 (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}[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   \fi
80 }
```

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`

```
81 \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](#).

```

82 \define@key{glossaries.sty}{style}{%
83   \renewcommand*{\@glossary@default@style}{#1}%
84 }

```

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
85 \newcommand*{\@gls@declareoption}[2]{%
86   \DeclareOptionX{#1}{#2}%
87   \DeclareOption{#1}{#2}%
88 }

```

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
89 \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).

```

90 \@gls@declareoption{nonumberlist}{%
91   \renewcommand*{\glossaryentrynumbers}[1]{\gls@save@numberlist{#1}}%
92 }

```

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

```

93 \define@boolkey{glossaries.sty}[gls]{savenumberlist}[true]{}
94 \gls@savenumberlistfalse

```

```

\@glo@seeautonumberlist
95 \newcommand*{\@glo@seeautonumberlist}{}

```

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

```

96 \@gls@declareoption{seeautonumberlist}{%
97   \renewcommand*{\@glo@seeautonumberlist}{%
98     \def\@glo@prefix{\glsnextpages}%
99   }%
100 }

```

`\@gls@loadlong`

```

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

```

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

```
102 \@gls@declareoption{nolong}{\renewcommand*{\@gls@loadlong}{}}
```

`\@gls@loadsuper` The package isn't loaded if isn't installed.

```
103 \IfFileExists{supertabular.sty}{%
104   \newcommand*{\@gls@loadsuper}{\RequirePackage{glossary-super}}}%
105   \newcommand*{\@gls@loadsuper}{}}
```

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

```
106 \@gls@declareoption{nosuper}{\renewcommand*{\@gls@loadsuper}{}}
```

`\@gls@loadlist`

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

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

```
108 \@gls@declareoption{nolist}{\renewcommand*{\@gls@loadlist}{}}
```

`\@gls@loadtree`

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

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

```
110 \@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).

```
111 \@gls@declareoption{nostyles}{%
112   \renewcommand*{\@gls@loadlong}{}%
113   \renewcommand*{\@gls@loadsuper}{}%
114   \renewcommand*{\@gls@loadlist}{}%
115   \renewcommand*{\@gls@loadtree}{}%
116   \let\@glossary@default@style\relax
117 }
```

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

```
118 \newcommand*{\glspostdescription}{%
119   \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi
120 }
```


nopostdot Boolean option to suppress post description dot

```

121 \define@boolkey{glossaries.sty}[gls]{nopostdot}[true]{}
122 \glsnopostdotfalse

```

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

```

123 \define@boolkey{glossaries.sty}[gls]{nogroupskip}[true]{}
124 \glsnogroupskipfalse

```

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

```

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

126 \@ifclassloaded{memoir}
127 {%
128   \glsucmarktrue
129 }%
130 {%
131   \glsucmarkfalse
132 }

```

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`.

```

133 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
134 \glsentrycounterfalse

```

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

```

135 \define@key{glossaries.sty}{counterwithin}{%
136   \renewcommand*{\@gls@counterwithin}{#1}%
137   \glsentrycountertrue
138 }

```

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

```

139 \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`.

```

140 \define@boolkey{glossaries.sty}[gls]{subentrycounter}[true]{}
141 \glssubentrycounterfalse

```

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

```

142 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
143   \csname @gls@setupsort@#1\endcsname
144 }

```

`\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.

```
145 \newcommand*\glsprestandardsort}[3]{%
146   \glsdosanitizesort
147 }
```

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

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

Store entry information when it's defined.

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

No count register required for standard sort.

```
150   \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.)

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

```
152     \ifx\@glo@sort\@glsdefaultsort
```

```
153       \let\@glo@sort\@glo@name
```

```
154     \fi
```

```
155     \let\glsdosanitizesort\@gls@sanitizesort
```

```
156     \glsprestandardsort{\@glo@sort}{##1}{##2}%
```

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

```
158   }%
```

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

```
159   \def\@gls@setsort##1{%
```

```
160 }
```

Set standard sort as the default:

```
161 \@gls@setupsort@standard
```

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

```
162 \newcommand*\glssortnumberfmt[1]{%
```

```
163   \ifnum#1<100000 0\fi
```

```
164   \ifnum#1<10000 0\fi
```

```
165   \ifnum#1<1000 0\fi
```

```
166   \ifnum#1<100 0\fi
```

```
167   \ifnum#1<10 0\fi
```

```
168   \number#1%
```

```
169 }
```

```

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

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

```

Set index information for this entry

```

200 \edef\@glo@type{\csname glo###1@type\endcsname}%
201 \edef\@gls@tmp{\csname glo###1@sort\endcsname}%
202 \ifx\@gls@tmp\@empty
203   \expandafter\global\expandafter
204   \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
205   \expandafter\protected@xdef\csname glo###1@sort\endcsname{%
206     \expandafter\glssortnumberfmt
207     {\csname glossary@\@glo@type @sortcount\endcsname}}%
208   \@glo@storeentry{##1}%
209 \fi
210 }%
211 }

```

`\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.)

```

212 \newcommand*{\glsdefmain}{%
213   \if@gls@docloaded
214     \newglossary[glg2]{main}{gls2}{glo2}{\glossaryname}%
215   \else
216     \newglossary{main}{gls}{glo}{\glossaryname}%
217   \fi
218 }

```

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`

```

219 \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`

```

220 \newcommand*{\acronymtype}{\glsdefaulttype}

```

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

```

221 \@gls@declareoption{nomain}{%
222   \let\glsdefaulttype\relax
223   \renewcommand*{\glsdefmain}{}%
224 }

```

`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.

```

225 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
226   \ifglsacronym
227     \renewcommand{\@gls@do@acronymsdef}{%
228       \DeclareAcronymList{acronym}%
229       \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
230       \renewcommand*{\acronymtype}{acronym}%
231     }%
232   \else
233     \let\@gls@do@acronymsdef\relax
234   \fi
235 }
```

`\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.

```

236 \AtBeginDocument{%
237   \ifglsacronym
238     \ifbool{glscompatible-3.07}%
239     {%
240     {%
241       \providecommand*{\printacronyms}[1][{}]{%
242         \printglossary[type=\acronymtype,#1]}%
243     }%
244   \fi
245 }
```

`@gls@do@acronymsdef` Set default value

```

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

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

```

247 \@gls@declareoption{acronyms}{%
248   \glsacronymtrue
249   \renewcommand{\@gls@do@acronymsdef}{%
250     \DeclareAcronymList{acronym}%
251     \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
252     \renewcommand*{\acronymtype}{acronym}%
253   }%
254 }
```

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

```

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

`\@addtoacronymlists`

```
256 \newcommand*{\@addtoacronymlists}[1]{%
257   \ifx\@glsacronymlists\@empty
258     \protected@xdef\@glsacronymlists{#1}%
259   \else
260     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
261   \fi
262 }
```

`\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.)

```
263 \newcommand*{\DeclareAcronymList}[1]{%
264   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
265 }
```

`\glsIfListOfAcronyms`

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

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

```
266 \newcommand{\glsIfListOfAcronyms}[1]{%
267   \edef\@do@gls@islistofacronyms{%
268     \noexpand\@gls@islistofacronyms{#1}{\@glsacronymlists}}%
269   \@do@gls@islistofacronyms
270 }
```

Internal command requires label and list to be expanded:

```
271 \newcommand{\@gls@islistofacronyms}[4]{%
272   \def\gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
273     \def\@before{##1}\def\@after{##2}}%
274   \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
275   \ifx\@after\@nnil
```

Not found

```
276     #4%
277   \else
```

Found

```
278     #3%
279   \fi
280 }
```

`\if@glsisacronymlist` Convenient boolean.

```
281 \newif\if@glsisacronymlist
```

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

```
282 \newcommand*{\gls@checkisacronymlist}[1]{%
283   \glsIfListOfAcronyms{#1}%
```

```

284     {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
285 }

```

`\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.)

```

286 \newcommand*\SetAcronymLists[1]{%
287   \renewcommand*\@glsacronymlists{#1}%
288 }

```

`acronymlists`

```

289 \define@key{glossaries.sty}{acronymlists}{%
290   \DeclareAcronymList{#1}%
291 }

```

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`

```

292 \newcommand{\glscounter}{page}

```

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

```

293 \define@key{glossaries.sty}{counter}{%
294   \renewcommand*\glscounter{#1}%
295 }

```

`\@gls@nohyperlist`

```

296 \newcommand*\@gls@nohyperlist{}

```

`\DeclareNoHyperList`

```

297 \newcommand*\GlsDeclareNoHyperList[1]{%
298   \ifdefempty\@gls@nohyperlist
299   {%
300     \renewcommand*\@gls@nohyperlist{#1}%
301   }%
302   {%
303     \appto\@gls@nohyperlist{,#1}%
304   }%
305 }

```

`nohypertypes`

```

306 \define@key{glossaries.sty}{nohypertypes}{%
307   \GlsDeclareNoHyperList{#1}%
308 }

```

`\GlossariesWarning` Prints a warning message.

```

309 \newcommand*{\GlossariesWarning}[1]{%
310   \PackageWarning{glossaries}{#1}%
311 }
```

`sariesWarningNoLine` Prints a warning message without the line number.

```

312 \newcommand*{\GlossariesWarningNoLine}[1]{%
313   \PackageWarningNoLine{glossaries}{#1}%
314 }
```

`nowarn` Define package option to suppress warnings

```

315 \@gls@declareoption{nowarn}{%
316   \renewcommand*{\GlossariesWarning}[1]{}%
317   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
318 }
```

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`

```

319 \newcommand*{\@gls@sanitizedesc}{%
320 }
321 %\end{macro}
322 %
323 %\begin{macro}{\glssetexpandfield}
324 %\changes{3.13a}{2013-11-05}{new}
325 %\begin{definition}
326 %\cs{glssetexpandfield}\marg{field}
327 %\end{definition}
328 % Sets field to always expand.
329 %   \begin{macrocode}
330 \newcommand*{\glssetexpandfield}[1]{%
331   \csdef{gls@assign@#1@field}##1##2{%
332     \@gls@expand@field{##1}{#1}{##2}%
333   }%
334 }
```

`\glssetnoexpandfield` `\glssetnoexpandfield{<field>}`

Sets field to never expand.

```

335 \newcommand*{\glssetnoexpandfield}[1]{%
336   \csdef{gls@assign@#1@field}##1##2{%
337     \@gls@noexpand@field{##1}{#1}{##2}%
338   }%
339 }
```


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

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

gn@descplural@field
 342 \glssetnoexpandfield{descplural}

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

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

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

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

@symbolplural@field
 347 \glssetnoexpandfield{symbolplural}

Sanitizing stuff:

\@gls@sanitizesort
 348 \newcommand*{\@gls@sanitizesort}{%
 349 \ifglssanitizesort
 350 \@onelevel@sanitize\@glo@sort
 351 \else
 352 \fi
 353 }

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.

```
354 \define@boolkey[gls]{sanitize}{description}[true]{%
355   \GlossariesWarning{sanitize={description} package option deprecated}%
356   \ifgls@sanitize@description
357     \glssetnoexpandfield{desc}%
358     \glssetnoexpandfield{descplural}%
359   \else
360     \glssetexpandfield{desc}%
361     \glssetexpandfield{descplural}%
362   \fi
363 }
```

```

364 \define@boolkey[glS]{sanitize}{name}[true]{%
365   \GlossariesWarning{sanitize={name} package option
366 deprecated}%
367   \ifglS@sanitize@name
368     \glSsetnoexpandfield{name}%
369   \else
370     \glSsetexpandfield{name}%
371   \fi
372 }

373 \define@boolkey[glS]{sanitize}{symbol}[true]{%
374   \GlossariesWarning{sanitize={symbol} package option
375 deprecated}%
376   \ifglS@sanitize@symbol
377     \glSsetnoexpandfield{symbol}%
378     \glSsetnoexpandfield{symbolplural}%
379   \else
380     \glSsetexpandfield{symbol}%
381     \glSsetexpandfield{symbolplural}%
382   \fi
383 }

```

sanitizesort

```

384 \define@boolkey{glossaries.sty}[glS]{sanitizesort}[true]{%
385   \ifglssanitizesort
386     \glSsetnoexpandfield{sortvalue}%
387   \else
388     \glSsetexpandfield{sortvalue}%
389   \fi
390 }

```

Default setting:

```

391 \glssanitizesorttrue
392 \glSsetnoexpandfield{sortvalue}%

393 \define@choicekey{sanitize}{sort}{true,false}[true]{%
394   \setbool{glssanitizesort}{#1}%
395   \ifglssanitizesort
396     \glSsetnoexpandfield{sortvalue}%
397   \else
398     \glSsetexpandfield{sortvalue}%
399   \fi
400   \GlossariesWarning{sanitize={sort} package option
401     deprecated. Use sanitizesort instead}%
402 }

```

sanitize

```

403 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
404 name=true]{%
405   \ifthenelse{\equal{#1}{none}}{}

```

```

406  {%
407    \GlossariesWarning{sanitize package option deprecated}%
408  }%
409  {%
410    \setkeys[glS]{sanitize}{#1}%
411  }%
412 }

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

ls@nottranslatorhook
414 \newcommand*{\@glS@nottranslatorhook{}}

nottranslate Provide a synonym for translate=false that can be passed via the document
              class.
415 \@glS@declareoption{nottranslate}{%
416   \glstranslatefalse
417   \let\@glS@nottranslatorhook\relax
418 }

translate Define translate option. If false don't set up multi-lingual support.
419 \define@choicekey{glossaries.sty}{translate}[\val\nr]%
420 {true,false,babel}[true]%
421 {%
422   \ifcase\nr\relax
423     \glstranslatetrue
424   \or
425     \glstranslatefalse
426     \let\@glS@nottranslatorhook\relax
427   \or
428     \glstranslatefalse
429     \def\@glS@nottranslatorhook{\RequirePackage{glossaries-babel}}%
430   \fi
431 }

Set the default value:
432 \glstranslatefalse
433 \@ifpackageloaded{translator}%
434   {\glstranslatetrue}%
435   {%
436     \@ifpackageloaded{polyglossia}%
437       {\glstranslatetrue}%
438       {%
439         \@ifpackageloaded{babel}{\glstranslatetrue}{}}%
440       }%
441 }

```

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

```
442 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
443 \glsexindexonlyfirstfalse
```

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

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

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

```
446 \newcommand*{\@gls@setacrstyle}{}%
```

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

```
447 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{}%
448 \ifbool{glsacrdescription}%
449 {}%
450 {}%
451 \renewcommand*{\@gls@sanitizedesc}{}%
452 }%
453 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
454 }
```

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

```
455 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{}%
456 \renewcommand*{\@gls@sanitizesymbol}{}%
457 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
458 }
```

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

```
459 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{}%
460 \renewcommand*{\@gls@sanitizesymbol}{}%
461 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
462 }
```

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

```
463 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{}%
464 \renewcommand*{\@gls@sanitizesymbol}{}%
465 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
466 }
```

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

```
467 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{}%
468 \renewcommand*{\@gls@sanitizesymbol}{}%
469 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
470 }
```

`shortcuts` Define acronym shortcuts.
471 `\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.
472 `\newcommand*{\glsorder}{word}`

`\@glsorder` The ordering information is written to the auxiliary file for `makeglossaries`, so ignore the auxiliary information.
473 `\newcommand*{\@glsorder}[1]{}`

`order`
474 `\define@choicekey{glossaries.sty}{order}{word,letter}{%`
475 `\def\glsorder{#1}}`

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

The default is `makeindex`:
477 `\glxindyfalse`

`makeindex` Define package option to specify that `makeindex` will be used to sort the glossaries:
478 `\@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.
479 `\define@boolkey[gls]{xindy}{glsnumbers}[true]{}`
480 `\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.)
481 `\def\@xdy@main@language{\language}%`

Define key to set the language
482 `\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.
483 `\ifcsundef{inputencodingname}{%`
484 `\def\gls@codepage{}}{%`
485 `\def\gls@codepage{inputencodingname}`
486 `}`

Define a key to set the code page.

```
487 \define@key[glS]{xindy}{codepage}{\def\glS@codepage{#1}}
```

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

```
488 \define@key{glossaries.sty}{xindy}[]{%
489   \glSxindytrue
490   \setkeys[glS]{xindy}{#1}%
491 }
```

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

```
492 \@glS@declareoption{xindygloss}{%
493   \glSxindytrue
494 }
```

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

```
495 \define@boolkey{glossaries.sty}[glS]{savewrites}[true]{%
496   \ifglSsavewrites
497     \renewcommand*{\glSwritefiles}{\@glSwritefiles}%
498   \else
499     \let\glSwritefiles\relax
500   \fi
501 }
```

Set default:

```
502 \glSsavewritesfalse
503 \let\glSwritefiles\relax
```

compatible-3.07

```
504 \define@boolkey{glossaries.sty}[glS]{compatible-3.07}[true]{%
505 \boolfalse{glScompatible-3.07}}
```

compatible-2.07

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

Also set 3.07 compatibility if this option is set.

```
507   \ifbool{glScompatible-2.07}%
508   {%
509     \booltrue{glScompatible-3.07}%
510   }%
511   {}%
512 }
513 \boolfalse{glScompatible-2.07}
```

symbols Create a “symbols” glossary type

```
514 \newcommand{\@glS@do@symbolsdef}{}
515 \@glS@declareoption{symbols}{%
516   \renewcommand{\@glS@do@symbolsdef}{%
```

```

517 \newglossary[slg]{symbols}{sls}{slo}{\glssymbolsgroupname}%
518 \newcommand*{\printsymbols}[1][\printglossary[type=symbols,#1]]%
519 }%
520 }

```

numbers Create a “symbols” glossary type

```

521 \newcommand{\@gls@do@numbersdef}{}
522 \@gls@declareoption{numbers}{%
523 \renewcommand{\@gls@do@numbersdef}{%
524 \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%
525 \newcommand*{\printnumbers}[1][\printglossary[type=numbers,#1]]%
526 }%
527 }

```

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

```

528 \@for\CurrentOption := \@declaredoptions\do{%
529 \ifx\CurrentOption\@empty
530 \else
531 \@expandtwoargs
532 \in@ {,\CurrentOption ,}{,\@classoptionslist,\@curroptions,}%
533 \ifin@
534 \@use@option
535 \expandafter \let\csname ds@\CurrentOption\endcsname\@empty
536 \fi
537 \fi
538 }

```

Now process options passed to the package:

```
539 \ProcessOptionsX
```

Load backward compatibility stuff:

```
540 \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:

```

541 \disable@keys{glossaries.sty}{compatible-2.07,%
542 xindy,xindygloss,makeindex,%
543 acronym,translate,nottranslate,nolong,nosuper,notree,nostyles,nomain}

```

Now define \setupglossaries:

```

544 \newcommand*{\setupglossaries}[1]{%
545 \renewcommand*{\@gls@setacrstyle}{}%
546 \ifglsacrshortcuts
547 \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
548 \else
549 \def\@gls@setupshortcuts{%
550 \ifglsacrshortcuts
551 \DefineAcronymSynonyms
552 \fi

```

```

553 }%
554 \fi
555 \glsacrshortcutsfalse
556 \let\@gls@do@numbersdef\relax
557 \let\@gls@do@symbolssdef\relax
558 \let\@gls@do@acronymsdef\relax
559 \setkeys{glossaries.sty}{#1}%
560 \@gls@setacrstyle
561 \@gls@setupshortcuts
562 \@gls@do@acronymsdef
563 \@gls@do@numbersdef
564 \@gls@do@symbolssdef
565 }

```

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

```

566 \ifglstranslate
567   \ifpackageloaded{polyglossia}%
568   {%
    polyglossia fakes babel so need to check for polyglossia first.
569   }%
570   {%
571     \@ifpackageloaded{babel}%
572     {%
573       \IfFileExists{translator.sty}%
574       {%
575         \RequirePackage{translator}%
576       }%
577     }%
578   }%
579   {}
580 }
581 \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}`).

```

582 \ifthenelse{\equal{\glscounter}{section}}{%
583 {%
584   \ifcsundef{chapter}{}%
585   {%
586     \let\@gls@old@chapter\@chapter

```



```

587 \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
588 \ifcsundef{hyperdef}}{\hyperdef{section}{\thesection}}}%
589 }%
590 }%
591 {}

```

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

```

592 \newcommand*{\@gls@onlypremakeg}{}

```

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

```

593 \newcommand*{\@onlypremakeg}[1]{%
594 \ifx\@gls@onlypremakeg\@empty
595 \def\@gls@onlypremakeg{#1}%
596 \else
597 \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
598 \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
599 \fi
600 }

```

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

```

601 \newcommand*{\@disable@onlypremakeg}{%
602 \for\@thiscs:=\@gls@onlypremakeg\do{%
603 \expandafter\@disable@premakecs\@thiscs%
604 }}

```

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

```

605 \newcommand*{\@disable@premakecs}[1]{%
606 \def#1{\PackageError{glossaries}{\string#1\space may only be
607 used before \string\makeglossaries}{You can't use
608 \string#1\space after \string\makeglossaries}}%
609 }

```

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`

```

610 \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`

```

611 \providecommand*{\acronymname}{Acronyms}

```

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

```
612 \newcommand*{\glsettoctitle}[1]{%
613 \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`

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

`\descriptionname`

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

`\symbolname`

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

`\pagelistname`

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

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

`glssymbolsgroupname`

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

`glnumbersgroupname`

```
619 \providecommand*{\glnumbersgroupname}{Numbers}
```

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

```
620 \newcommand*{\glpluralsuffix}{s}
```

`\seename`

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

`\andname`

```
622 \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.)

```
623 \newcommand*{\addglossarytocaptions}[1]{%
624   \ifcsundef{captions#1}{}%
625   {%
626     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
```

```

627 \expandafter\toks@\expandafter{\@gls@tmp
628 \renewcommand*\glossaryname{\translate{Glossary}}}%
629 }%
630 \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
631 }%
632 }

633 \ifglstranslate

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

634 \@ifpackageloaded{translator}{%
635 \usedictionary{glossaries-dictionary}%
636 \addglossarytocaptions{portuges}%
637 \addglossarytocaptions{portuguese}%
638 \addglossarytocaptions{brazil}%
639 \addglossarytocaptions{brazilian}%
640 \addglossarytocaptions{danish}%
641 \addglossarytocaptions{dutch}%
642 \addglossarytocaptions{afrikaans}%
643 \addglossarytocaptions{english}%
644 \addglossarytocaptions{UKenglish}%
645 \addglossarytocaptions{USenglish}%
646 \addglossarytocaptions{american}%
647 \addglossarytocaptions{australian}%
648 \addglossarytocaptions{british}%
649 \addglossarytocaptions{canadian}%
650 \addglossarytocaptions{newzealand}%
651 \addglossarytocaptions{french}%
652 \addglossarytocaptions{frenchb}%
653 \addglossarytocaptions{francais}%
654 \addglossarytocaptions{acadian}%
655 \addglossarytocaptions{canadien}%
656 \addglossarytocaptions{german}%
657 \addglossarytocaptions{germanb}%
658 \addglossarytocaptions{austrian}%
659 \addglossarytocaptions{naustrian}%
660 \addglossarytocaptions{ngerman}%
661 \addglossarytocaptions{irish}%
662 \addglossarytocaptions{italian}%
663 \addglossarytocaptions{magyar}%
664 \addglossarytocaptions{hungarian}%
665 \addglossarytocaptions{polish}%
666 \addglossarytocaptions{spanish}%
667 \renewcommand*\glssettoctitle}[1]{%
668 \ifthenelse{\equal{#1}{main}}{%
669 \translatelet{\glossarytoctitle}{Glossary}}{%
670 \ifthenelse{\equal{#1}{acronym}}{%
671 \translatelet{\glossarytoctitle}{Acronyms}}{%
672 \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
673 \renewcommand*\glossaryname{\translate{Glossary}}%

```

```

674 \renewcommand*{\acronymname}{\translate{Acronyms}}}%
675 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}}%
676 \renewcommand*{\descriptionname}{%
677 \translate{Description (glossaries)}}}%
678 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}}%
679 \renewcommand*{\pagelistname}{%
680 \translate{Page List (glossaries)}}}%
681 \renewcommand*{\glssymbolsgroupname}{%
682 \translate{Symbols (glossaries)}}}%
683 \renewcommand*{\glsnumbersgroupname}{%
684 \translate{Numbers (glossaries)}}}%
685 }{%

686 \@ifpackageloaded{polyglossia}%
687 {\RequirePackage{glossaries-polyglossia}}%
688 {%
689 \@ifpackageloaded{babel}{%
690 \RequirePackage{glossaries-babel}}}%
691 }}
692 \else

693 \@gls@notranslatorhook
694 \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.

```
695 \DeclareRobustCommand*{\nopostdesc}{}

```

`\@nopostdesc` Suppress next description terminator.

```

696 \newcommand*{\@nopostdesc}{%
697 \let\org@glspostdescription\glspostdescription
698 \def\glspostdescription{%
699 \let\glspostdescription\org@glspostdescription}%
700 }

```

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

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

```

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

```
702 \newcommand{\glspar}{\par}

```

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

```

703 \ifglxindy
704 \newcommand{\setStyleFile}[1]{%
705 \renewcommand{\istfilename}{#1.xdy}}
706 \else
707 \newcommand{\setStyleFile}[1]{%
708 \renewcommand{\istfilename}{#1.ist}}
709 \fi

```

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

```
710 \@onlypremakeg\setStyleFile
```

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

`\istfilename`

```
711 \ifglxindy
712   \def\istfilename{\jobname.xdy}
713 \else
714   \def\istfilename{\jobname.ist}
715 \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 it is not required by \TeX , `\@istfilename` ignores its argument.

`\@istfilename`

```
716 \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`

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

`\glsSetCompositor` Sets the compositor.

```
718 \newcommand*{\glsSetCompositor}[1]{%
719   \renewcommand*{\glscompositor}{#1}}
```

Only use before `\makeglossaries`

```
720 \@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`.

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

`\glsSetAlphaCompositor` Sets the alpha compositor.

```
722 \ifglxsindy
723   \newcommand*\glsSetAlphaCompositor[1]{%
724     \renewcommand*\@glsAlphacompositor{#1}}
725 \else
726   \newcommand*\glsSetAlphaCompositor[1]{%
727     \glsnxindywarning\glsSetAlphaCompositor}
728 \fi
```

Can only be used before `\makeglossaries`

```
729 \@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.

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

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

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

Only has an effect when used before `\makeglossaries`

```
733 \@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.

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

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

```
735 \newcommand*\{\glsSetSuffixFF}[1]{%
736   \renewcommand*\{\gls@suffixFF}{#1}%
737 }
```

`\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”.

```
738 \ifcsundef{hyperlink}%
739 {%
740   \newcommand*\{\glsnumberformat}[1]{#1}%
741 }%
742 {%
743   \newcommand*\{\glsnumberformat}[1]{\glshypernumber{#1}}%
744 }
```

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`

```
745 \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`

```
746 \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`

```
747 \newcommand*{\glossarypreamble}{%
748   \csuse{@glossarypreamble@currentglossary}%
749 }
```

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

Code provided by Michael Pock.

```
750 \newcommand{\setglossarypreamble}[2][\glsdefaulttype]{%
751   \ifglossaryexists{#1}{%
752     \csgdef{@glossarypreamble@#1}{#2}%
753   }{%
754     \GlossariesWarning{%
755       Glossary ‘#1’ is not defined%
756     }%
757   }%
758 }
```

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
759 \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.
760 \newcommand*{\glossarysection}[2][\@gls@title]{%
761   \def\@gls@title{#2}%
762   \ifcsundef{phantomsection}%
763   {%
764     \@glossarysection{#1}{#2}%
765   }%
766   {%
767     \p@glossarysection{#1}{#2}%
768   }%

769   \glsglossarymark{\glossarytoctitle}%
770 }

\glsglossarymark Sets the header mark for the glossary. Takes the glossary short (TOC) title as the
                  argument.
771 \ifcsundef{glossarymark}%
772 {%
773   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}
774 }%
775 {%
776   \@ifclassloaded{memoir}
777   {%
778     \newcommand{\glsglossarymark}[1]{%
779       \ifglsucmark
780         \markboth{\memUHead{#1}}{\memUHead{#1}}%
781       \else
782         \markboth{#1}{#1}%
783       \fi
784     }
785   }%
786   {%
787     \newcommand{\glsglossarymark}[1]{%
788       \ifglsucmark
789         \mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
790       \else
791         \mkboth{#1}{#1}%
792       \fi
793     }
794   }
795 }

\glossarymark Provided for backward compatibility:

```



```

796 \providecommand{\glossarymark}[1]{%
797   \ifglsucmark
798     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
799   \else
800     \@mkboth{#1}{#1}%
801   \fi
802 }

```

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`

```

803 \newcommand*\setglossarysection[1]{%
804 \setkeys{glossaries.sty}{section=#1}}

```

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

`\@glossarysection`

```

805 \newcommand*\@glossarysection[2]{%
806   \ifx\@glossarysecstar\@empty
807     \csname\@glossarysec\endcsname{#2}%
808   \else
809     \csname\@glossarysec\endcsname*{#2}%
810     \@gls@toc{#1}{\@glossarysec}%
811   \fi
812   \@glossaryseclabel
813 }

```

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`

```

814 \newcommand*\@pglossarysection[2]{%
815   \glsclearpage
816   \phantomsection
817   \ifx\@glossarysecstar\@empty
818     \csname\@glossarysec\endcsname{#2}%
819   \else
820     \@gls@toc{#1}{\@glossarysec}%
821     \csname\@glossarysec\endcsname*{#2}%
822   \fi
823   \@glossaryseclabel
824 }

```

`\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.

```

825 \newcommand*{\gls@doclearpage}{%
826   \ifthenelse{\equal{\@glossarysec}{chapter}}{%
827     {%
828       \ifcsundef{cleardoublepage}%
829       {%
830         \clearpage
831       }%
832     }%
833     \ifcsdef{if@openright}%
834     {%
835       \if@openright
836         \cleardoublepage
837       \else
838         \clearpage
839       \fi
840     }%
841     {%
842       \cleardoublepage
843     }%
844   }%
845 }%
846 {}%
847 }

```

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

```

848 \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`

```

849 \newcommand*{\@gls@toc}[2]{%
850   \ifglstoc
851     \ifglslnumberline
852       \addcontentsline{toc}{#2}{\numberline{#1}}%
853     \else
854       \addcontentsline{toc}{#2}{#1}%
855     \fi
856   \fi
857 }

```

1.4 Xindy

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

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

```
858 \newcommand*{\glsnnoxindywarning}[1]{%
859   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
860 }
```

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

```
861 \ifglsxindy
862   \edef\@xdyattributes{\string"default\string"}%
863 \fi
```

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

```
864 \ifglsxindy
865   \edef\@xdyattributelist{}%
866 \fi
```

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

```
867 \ifglsxindy
868   \def\@xdylocref{}
869 \fi
```

`\@gls@ifinlist`

```
870 \newcommand*{\@gls@ifinlist}[4]{%
871   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
872     \def\@gls@listsuffix{##2}%
873     \ifx\@gls@listsuffix\@empty
874       #4%
875     \else
876       #3%
877     \fi
878   }%
879   \@do@ifinlist,#2,#1,\end@do@ifinlist
880 }
```

`\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.

```
881 \ifglsxindy
882   \newcommand*{\@xdycounters}{\glscounter}
883   \newcommand*\GlsAddXdyCounters[1]{%
884     \@for\@gls@ctr:=#1\do{%
```

Check if already in list before adding.

```

885     \edef\@do@addcounter{%
886         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
887         {%
888             \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
889                 \noexpand\@gls@ctr}%
890         }%
891     }%
892     \@do@addcounter
893 }
894 }

```

Only has an effect before `\writeist`:

```

895 \@onlypremakeg\GlsAddXdyCounters
896 \else
897 \newcommand*\GlsAddXdyCounters[1]{%
898     \glsnoxindywarning\GlsAddXdyAttribute
899 }
900 \fi

```

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

```

901 \newcommand*\@disabled@glssaddxdycounters{%
902     \PackageError{glossaries}{\string\GlsAddXdyCounters\space
903     can't be used after \string\GlsAddXdyAttribute}{Move all
904     occurrences of \string\GlsAddXdyCounters\space before the first
905     instance of \string\GlsAddXdyAttribute}%
906 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

907 \ifglsxindy

```

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

```

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

```

Add to xindy attribute list

```

909     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
910         \string"#2#1\string"}%

```

Add to xindy markup location.

```

911     \expandafter\toks@\expandafter{\@xdylocref}%
912     \edef\@xdylocref{\the\toks@ ^^J%
913         (markup-locref
914         :open \string"\string~n%
915         \expandafter\string\csname glsX#2X#1\endcsname
916         \string" ^^J
917         :close \string"\string" ^^J
918         :attr \string"#2#1\string")}%

```

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

```

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

```

```

920     \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
921 }%
922 }

```

High-level command:

```

923 \newcommand*\GlsAddXdyAttribute[1]{%

```

Add to comma-separated attribute list

```

924 \ifx\@xdyattributelist\@empty
925 \edef\@xdyattributelist{#1}%
926 \else
927 \edef\@xdyattributelist{\@xdyattributelist,#1}%
928 \fi

```

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

```

929 \@for\@this@counter:=\@xdycounters\do{%
930 \protected@edef\gls@do@addxdyattribute{%
931 \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
932 }
933 \gls@do@addxdyattribute
934 }%

```

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

```

935 \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
936 }

```

Only has an effect before `\writeist`:

```

937 \@onlypremakeg\GlsAddXdyAttribute
938 \else
939 \newcommand*\GlsAddXdyAttribute[1]{%
940 \glsnoxindywarning\GlsAddXdyAttribute}
941 \fi

```

redefinedattributes Add known attributes for all defined counters

```

942 \ifglxsindy
943 \newcommand*\@gls@addpredefinedattributes{%
944 \GlsAddXdyAttribute{glsnumberformat}
945 \GlsAddXdyAttribute{textrm}
946 \GlsAddXdyAttribute{textsf}
947 \GlsAddXdyAttribute{texttt}
948 \GlsAddXdyAttribute{textbf}
949 \GlsAddXdyAttribute{textmd}
950 \GlsAddXdyAttribute{textit}
951 \GlsAddXdyAttribute{textup}
952 \GlsAddXdyAttribute{textsl}
953 \GlsAddXdyAttribute{textsc}
954 \GlsAddXdyAttribute{emph}
955 \GlsAddXdyAttribute{glshypernumber}
956 \GlsAddXdyAttribute{hyperrrm}
957 \GlsAddXdyAttribute{hypersf}
958 \GlsAddXdyAttribute{hypertt}

```

```

959 \GlsAddXdyAttribute{hyperbf}
960 \GlsAddXdyAttribute{hypermd}
961 \GlsAddXdyAttribute{hyperit}
962 \GlsAddXdyAttribute{hyperup}
963 \GlsAddXdyAttribute{hypersl}
964 \GlsAddXdyAttribute{hypersc}
965 \GlsAddXdyAttribute{hyperemph}
966 }
967 \else
968 \let\@gls@addpredefinedattributes\relax
969 \fi

```

`\@xdyuseralphabets` List of additional alphabets

```

970 \def\@xdyuseralphabets{}

```

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

```

971 \ifglsxindy
972 \newcommand*{\GlsAddXdyAlphabet}[2]{%
973 \edef\@xdyuseralphabets{%
974 \@xdyuseralphabets ^^J
975 (define-alphabet "#1" (#2))}}
976 \else
977 \newcommand*{\GlsAddXdyAlphabet}[2]{%
978 \glsnoxindywarning\GlsAddXdyAlphabet}
979 \fi

```

This code is only required for xindy:

```

980 \ifglsxindy

```

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

```

981 \newcommand*{\@gls@xdy@locationlist}{%
982 roman-page-numbers,%
983 Roman-page-numbers,%
984 arabic-page-numbers,%
985 alpha-page-numbers,%
986 Alpha-page-numbers,%
987 Appendix-page-numbers,%
988 arabic-section-numbers%
989 }

```

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.

```

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

```

```

991     \string"roman-numbers-lowercase\string" :sep \string"}}%
992 \@onelevel@sanitize\@gls@roman
993 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
994     :sep \string"}%
995 \@onelevel@sanitize\@tmp
996 \ifx\@tmp\@gls@roman
997     \expandafter
998     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
999         \string"roman-numbers-lowercase\string"%
1000     }%
1001 \else
1002     \expandafter
1003     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
1004         :sep \string"\@gls@roman\string"%
1005     }%
1006 \fi

```

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

```

1007 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
1008     \string"roman-numbers-uppercase\string"%
1009 }%

```

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

```

1010 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
1011     \string"arabic-numbers\string"%
1012 }%

```

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

```

1013 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
1014     \string"alpha\string"%
1015 }%

```

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

```

1016 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
1017     \string"ALPHA\string"%
1018 }%

```

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

```

1019 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1020     \string"ALPHA\string"
1021     :sep \string"\@glsAlphacompositor\string"
1022     \string"arabic-numbers\string"%
1023 }

```

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

```

1024 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%

```

```

1025   \string"arabic-numbers\string"
1026   :sep \string"\glscompositor\string"
1027   \string"arabic-numbers\string"%
1028   }%

```

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

```

1029   \def\xdyuserlocationdefs{}

```

`\xdyuserlocationnames` List of additional user location names

```

1030   \def\xdyuserlocationnames{}

```

End of xindy-only block:

```

1031 \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.)

```

1032 \ifglsxindy
1033   \newcommand*\GlsAddXdyLocation[3][{}]{%
1034     \def\@gls@tmp{#1}%
1035     \ifx\@gls@tmp\@empty
1036       \edef\xdyuserlocationdefs{%
1037         \xdyuserlocationdefs ^^J%
1038         (define-location-class \string"#2\string"^^J\space\space
1039         \space(:sep \string"{}\glsopenbrace\string" #3
1040         :sep \string"\glsclosebrace\string"))
1041       }%
1042     \else
1043       \edef\xdyuserlocationdefs{%
1044         \xdyuserlocationdefs ^^J%
1045         (define-location-class \string"#2\string"^^J\space\space
1046         \space(:sep "\glsopenbrace"
1047         #1
1048         :sep "\glsclosebrace\glsopenbrace" #3
1049         :sep "\glsclosebrace"))
1050       }%
1051     \fi
1052     \edef\xdyuserlocationnames{%
1053       \xdyuserlocationnames^^J\space\space\space
1054       \string"#1\string"}%
1055   }

```

Only has an effect before `\writeist`:

```

1056   \@onlypremake\GlsAddXdyLocation
1057 \else
1058   \newcommand*\GlsAddXdyLocation[2]{%
1059     \glsnnoxindywarning\GlsAddXdyLocation}
1060 \fi

```


ylocationclassorder Define location class order

```
1061 \ifglxindy
1062   \edef\@xdylocationclassorder{^^J\space\space\space
1063     \string"roman-page-numbers\string"^^J\space\space\space
1064     \string"arabic-page-numbers\string"^^J\space\space\space
1065     \string"arabic-section-numbers\string"^^J\space\space\space
1066     \string"alpha-page-numbers\string"^^J\space\space\space
1067     \string"Roman-page-numbers\string"^^J\space\space\space
1068     \string"Alpha-page-numbers\string"^^J\space\space\space
1069     \string"Appendix-page-numbers\string"
1070     \@xdyuserlocationnames^^J\space\space\space
1071     \string"see\string"
1072   }
1073 \fi
```

Change the location order.

yLocationClassOrder

```
1074 \ifglxindy
1075   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1076     \def\@xdylocationclassorder{#1}}
1077 \else
1078   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1079     \glsnxindywarning\GlsSetXdyLocationClassOrder}
1080 \fi
```

\@xdysortrules Define sort rules

```
1081 \ifglxindy
1082   \def\@xdysortrules{}
1083 \fi
```

\GlsAddSortRule Add a sort rule

```
1084 \ifglxindy
1085   \newcommand*\GlsAddSortRule[2]{%
1086     \expandafter\toks@\expandafter{\@xdysortrules}%
1087     \protected@edef\@xdysortrules{\the\toks@ ^^J
1088       (sort-rule \string"#1\string" \string"#2\string")}%
1089   }
1090 \else
1091   \newcommand*\GlsAddSortRule[2]{%
1092     \glsnxindywarning\GlsAddSortRule}
1093 \fi
```

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

```
1094 \ifglxindy
1095   \def\@xdyrequiredstyles{tex}
1096 \fi
```

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

```
1097 \ifglxindy
1098   \newcommand*\GlsAddXdyStyle[1]{%
1099     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
1100 \else
1101   \newcommand*\GlsAddXdyStyle[1]{%
1102     \glsnnoxindywarning\GlsAddXdyStyle}
1103 \fi
```

`\GlsSetXdyStyles` Reset the list of required styles

```
1104 \ifglxindy
1105   \newcommand*\GlsSetXdyStyles[1]{%
1106     \edef\@xdyrequiredstyles{#1}}
1107 \else
1108   \newcommand*\GlsSetXdyStyles[1]{%
1109     \glsnnoxindywarning\GlsSetXdyStyles}
1110 \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` no longer available. Now provide a command that does nothing (in case it's been patched).

```
1111 \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.

```
1112 \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.

```
1113 \ifglxindy
1114   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1115     \ifglossaryexists{#1}{%
1116       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1117     }{%
1118       \PackageError{glossaries}{Can't set language type for
1119         glossary type '#1' --- no such glossary}{%
1120         You have specified a glossary type that doesn't exist}}
1121 \else
1122   \newcommand*\GlsSetXdyLanguage[2][]{%
1123     \glsnnoxindywarning\GlsSetXdyLanguage}
1124 \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.

```
1125 \def\@gls@codepage#1#2{}
```

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

```
1126 \ifglxindy
1127   \newcommand*\GlsSetXdyCodePage[1]{%
1128     \renewcommand*\gls@codepage{#1}%
1129   }
```

Suggested by egreg:

```
1130   \AtBeginDocument{%
1131     \ifx\gls@codepage\@empty
1132       \@ifpackageloaded{fontspec}{\def\gls@codepage{utf8}}{}%
1133     \fi
1134   }
1135 \else
1136   \newcommand*\GlsSetXdyCodePage[1]{%
1137     \glsnoxindywarning\GlsSetXdyCodePage}
1138 \fi
```

`\@xdylettergroups` Store letter group definitions.

```
1139 \ifglxindy
1140   \ifglx@xindy@glsnumbers
1141     \def\@xdylettergroups{(define-letter-group
1142       \string\glsnumbers\string^^J\space\space\space
1143       :prefixes (\string"0\string" \string"1\string"
1144       \string"2\string" \string"3\string" \string"4\string"
1145       \string"5\string" \string"6\string" \string"7\string"
1146       \string"8\string" \string"9\string")^^J\space\space\space
1147       :before \string"@glsfirstletter\string")}
1148   \else
1149     \def\@xdylettergroups{}
1150   \fi
1151 \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.

```
1152   \newcommand*\GlsAddLetterGroup[2]{%
1153     \expandafter\toks@\expandafter{\@xdylettergroups}%
1154     \protected@edef\@xdylettergroups{\the\toks@^^J%
1155     (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
1156   }
```

1.5 Loops and conditionals

`\forallglossaries` 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.

```
1157 \newcommand*\forallglossaries[3][\@glo@types]{%
1158   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1159 }
```

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

`\forall glsentries[<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.

```
1160 \newcommand*\forallglsentries[3][\glsdefaulttype]{%
1161   \edef\@glo@list{\csname glolist@#1\endcsname}%
1162   \@for#2:=\@glo@list\do
1163   {%
1164     \ifdefempty{#2}{-}{#3}%
1165   }%
1166 }
```

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

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

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

```
1167 \newcommand*\forallallglsentries[3][\@glo@types]{%
1168   \expandafter\forallglossaries\expandafter[1]{\@this@glo@}%
1169   {%
1170     \forallglsentries[\@this@glo@]{#2}{#3}%
1171   }%
1172 }
```

`\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.

```
1173 \newcommand{\ifglossaryexists}[3]{%
1174   \ifcsundef{@glo@type@#1@out}{#3}{#2}%
1175 }
```

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

`\if glsentryexists{<label>}{<true text>}{<false text>}`

where $\langle label \rangle$ is the entry's label.

```
1176 \newcommand{\ifglstryexists}[3]{%
1177   \ifcsundef{glo@#1@name}{#3}{#2}%
1178 }
```

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

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

where $\langle label \rangle$ is the entry's label. If true it will do $\langle true text \rangle$ otherwise it will do $\langle false text \rangle$.

```
1179 \newcommand*{\ifglused}[3]{\ifbool{glo@#1@flag}{#2}{#3}}
```

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

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

Generate an error if entry specified by $\langle label \rangle$ doesn't exist, otherwise do $\langle code \rangle$.

```
1180 \newcommand{\glstryexists}[2]{%
1181   \ifglstryexists{#1}{#2}{%
1182     \PackageError{glossaries}{Glossary entry ‘#1’ has not been
1183     defined}{You need to define a glossary entry before you
1184     can use it.}}%
1185 }
```

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

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

```
1186 \newcommand{\glstrynoexists}[2]{%
1187   \ifglstryexists{#1}{%
1188     \PackageError{glossaries}{Glossary entry ‘#1’ has already
1189     been defined}{#2}%
1190 }
```

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

```
1191 \newcommand{\ifglshaschildren}[3]{%
1192   \glstryexists{#1}%
1193   {%
1194     \def\do@glshaschildren{#3}%
1195     \expandafter\forglstry\expandafter[\csname glo@#1@type\endcsname]
1196     {\glo@label}%
1197     {%
1198       \letcs\glo@parent{glo@\glo@label @parent}%
1199       \ifthenelse{\equal{#1}{\glo@parent}}{%
1200         {%
1201           \def\do@glshaschildren{#2}%
1202           \@endfortrue
```

```

1203     }%
1204     {}%
1205     }%
1206     \do@glshaschildren
1207 }%
1208 }

\ifglshasparent \ifglshaschildren{<label>}{<true part>}{<false part>}
1209 \newcommand{\ifglshasparent}[3]{%
1210   \glsdoifexists{#1}%
1211   {%
1212     \ifcseempty{glo@#1@parent}{#3}{#2}%
1213   }%
1214 }

\ifglshasdesc \ifglshasdesc{<label>}{<true part>}{<false part>}
1215 \newcommand*\ifglshasdesc[3]{%
1216   \ifcseempty{glo@#1@desc}%
1217   {#3}%
1218   {#2}%
1219 }

ifglsdessuppressed \ifglsdessuppressed{<label>}{<true part>}{<false part>} Does <true part>
if the description is just \nopostdesc otherwise does <false part>.
1220 \newcommand*\ifglsdessuppressed[3]{%
1221   \ifcsequal{glos@#1@desc}{@no@post@desc}%
1222   {#2}%
1223   {#3}%
1224 }

\ifglshassymbol \ifglshassymbol{<label>}{<true part>}{<false part>}
1225 \newcommand*\ifglshassymbol[3]{%
1226   \ifcseempty{glo@#1@symbol}%
1227   {#3}%
1228   {%
1229     \expandafter\ifx\csname glo@#1@symbol\endcsname\@gls@default@value
1230     #3%
1231   \else
1232     #2%
1233   \fi
1234 }%
1235 }

\ifglshaslong \ifglshaslong{<label>}{<true part>}{<false part>}
1236 \newcommand*\ifglshaslong[3]{%
1237   \ifcseempty{glo@#1@long}%
1238   {#3}%
1239   {%

```

```

1240 \expandafter\ifx\csname glo@#1@long\endcsname\@gls@default@value
1241 #3%
1242 \else
1243 #2%
1244 \fi
1245 }%
1246 }

```

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

```

1247 \newcommand*{\ifglshasshort}[3]{%
1248 \ifcsempy{glo@#1@short}%
1249 {#3}%
1250 {%
1251 \expandafter\ifx\csname glo@#1@short\endcsname\@gls@default@value
1252 #3%
1253 \else
1254 #2%
1255 \fi
1256 }%
1257 }

```

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`

```

1258 \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.

```

1259 \newcommand*{\@gls@provide@newglossary{%
1260 \protected@write\@auxout{}\string\providecommand\string\@newglossary[4]{}%
    Only need to do this once.
1261 \let\@gls@provide@newglossary\relax
1262 }

```

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

```

1263 \newcommand*{\defglentryfmt}[2][\glsdefaulttype]{%
1264 \csgdef{gls@#1@entryfmt}{#2}%
1265 }

```

`\gls@doentryfmt`

```

1266 \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`

```
1267 \newcommand*{\newglossary}[5][glg]{%
1268   \ifglossaryexists{#2}%
1269   {%
1270     \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
1271       You can’t define a new glossary called ‘#2’ because it already
1272       exists}%
1273   }%
1274   {%
      Check if default has been set
1275     \ifundef\glsdefaulttype
1276     {%
1277       \gdef\glsdefaulttype{#2}%
1278     }{}%
      Add this to the list of glossary types:
1279     \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.
1280     \expandafter\gdef\csname glolist@#2\endcsname{,}%
      Store details of this new glossary type:
1281     \expandafter\def\csname @glotype@#2in\endcsname{#3}%
1282     \expandafter\def\csname @glotype@#2out\endcsname{#4}%
1283     \expandafter\def\csname @glotype@#2title\endcsname{#5}%
1284     \@gls@provide@newglossary
1285     \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.


```

1286 \ifcsundef{gls@#2@entryfmt}%
1287 {%
1288     \defglsentryfmt[#2]{\glsentryfmt}%
1289 }%
1290 {}%

```

Define sort counter if required:

```

1291 \@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.)

```

1292 \@ifnextchar[{\@gls@setcounter{#2}}%
1293     {\@gls@setcounter{#2}[\glscounter]}}%
1294 }

```

`\altnewglossary`

```

1295 \newcommand*{\altnewglossary}[3]{%
1296     \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1297 }

```

Only define new glossaries in the preamble:

```

1298 \@onlypreamble{\newglossary}

```

Only define new glossaries before `\makeglossaries`

```

1299 \@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`

```

1300 \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`

```

1301 \def\@gls@setcounter#1[#2]{%
1302     \expandafter\def\csname @glotype@#1@counter\endcsname{#2}%

```

Add counter to xindy list, if not already added:

```

1303     \ifglxindy
1304         \GlsAddXdyCounters{#2}%
1305     \fi
1306 }

```

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

`\@gls@getcounter`

```

1307 \newcommand*{\@gls@getcounter}[1]{%
1308     \csname @glotype@#1@counter\endcsname
1309 }

```

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

```
1310 \glsdefmain
```

Define the “acronym” glossaries if required.

```
1311 \@gls@do@acronymsdef
```

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

```
1312 \@gls@do@symbolsdef
```

```
1313 \@gls@do@numbersdef
```

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.

```
1314 \define@key{glossentry}{name}{%
```

```
1315 \def\@glo@name{#1}}%
```

```
1316 }
```

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`.

```
1317 \define@key{glossentry}{description}{%
```

```
1318 \def\@glo@desc{#1}}%
```

```
1319 }
```

descriptionplural

```
1320 \define@key{glossentry}{descriptionplural}{%
```

```
1321 \def\@glo@descplural{#1}}%
```

```
1322 }
```

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`.

```
1323 \define@key{glossentry}{sort}{%
```

```
1324 \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.

```
1325 \define@key{glossentry}{text}{%
1326 \def\@glo@text{#1}%
1327 }
```

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.

```
1328 \define@key{glossentry}{plural}{%
1329 \def\@glo@plural{#1}%
1330 }
```

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.

```
1331 \define@key{glossentry}{first}{%
1332 \def\@glo@first{#1}%
1333 }
```

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.

```
1334 \define@key{glossentry}{firstplural}{%
1335 \def\@glo@firstplural{#1}%
1336 }
```

`\@gls@default@value`

```
1337 \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).

```
1338 \define@key{glossentry}{symbol}{%
1339 \def\@glo@symbol{#1}%
1340 }
```

symbolplural

```
1341 \define@key{glossentry}{symbolplural}{%
1342 \def\@glo@symbolplural{#1}%
1343 }
```

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

```
1344 \define@key{glossentry}{type}{%
1345 \def\@glo@type{#1}}
```

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

```
1346 \define@key{glossentry}{counter}{%
1347 \ifcsundef{c@#1}%
1348 {%
1349 \PackageError{glossaries}%
1350 {There is no counter called ‘#1’}%
1351 {%
1352 The counter key should have the name of a valid counter
1353 as its value%
1354 }%
1355 }%
1356 {%
1357 \def\@glo@counter{#1}%
1358 }%
1359 }
```

see The see key specifies a list of cross-references

```
1360 \define@key{glossentry}{see}{%
1361 \gls@checkseeallowed
1362 \def\@glo@see{#1}%
1363 \@glo@seeautonumberlist
1364 }
```

gls@checkseeallowed

```
1365 \newcommand*{\gls@checkseeallowed}{%
1366 \PackageError{glossaries}%
1367 {‘see’ key may only be used after \string\makeglossaries}%
1368 {You must use \string\makeglossaries\space before defining
1369 any entries that have a ‘see’ key}%
1370 }
```

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

```
1371 \define@key{glossentry}{parent}{%
1372 \def\@glo@parent{#1}}
```

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

```
1373 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1374 \ifcase\nr\relax
1375 \def\@glo@prefix{\glsnonextpages}%
1376 \else
1377 \def\@glo@prefix{\glsnextpages}%
1378 }
```

```
1378 \fi
1379 }
```

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

user1

```
1380 \define@key{glossentry}{user1}{%
1381 \def\@glo@useri{#1}%
1382 }
```

user2

```
1383 \define@key{glossentry}{user2}{%
1384 \def\@glo@userii{#1}%
1385 }
```

user3

```
1386 \define@key{glossentry}{user3}{%
1387 \def\@glo@useriii{#1}%
1388 }
```

user4

```
1389 \define@key{glossentry}{user4}{%
1390 \def\@glo@useriv{#1}%
1391 }
```

user5

```
1392 \define@key{glossentry}{user5}{%
1393 \def\@glo@userv{#1}%
1394 }
```

user6

```
1395 \define@key{glossentry}{user6}{%
1396 \def\@glo@uservi{#1}%
1397 }
```

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.

```
1398 \define@key{glossentry}{short}{%
1399 \def\@glo@short{#1}%
1400 }
```

shortplural This key is provided for use by \newacronym.

```
1401 \define@key{glossentry}{shortplural}{%
1402 \def\@glo@shortpl{#1}%
1403 }
```

long This key is provided for use by \newacronym.

```
1404 \define@key{glossentry}{long}{%
1405 \def\@glo@long{#1}%
1406 }
```

longplural This key is provided for use by \newacronym.

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

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

```
1410 \newcommand*{\@glsnoname}{%
1411   \PackageError{glossaries}{name key required in
1412     \string\newglossaryentry\space for entry '\@glo@label'}{You
1413     haven't specified the entry name}}
```

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

```
1414 \newcommand*{\@glsnodelsc}{%
1415   \PackageError{glossaries}
1416   {%
1417     description key required in \string\newglossaryentry\space
1418     for entry '\@glo@label'%
1419   }%
1420   {%
1421     You haven't specified the entry description%
1422   }%
1423 }
```

\@glsdefaultplural Now obsolete. Don't use.

```
1424 \newcommand*{\@glsdefaultplural}{{}}
```

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

```
1425 \newcommand*{\@gls@missingnumberlist}[1]{%
1426   ??%
1427   \ifglssavenumberlist
1428     \GlossariesWarning{Missing number list for entry '#1'.
1429       Maybe makeglossaries + rerun required.}%
1430   \else
1431     \PackageError{glossaries}%
1432     {Package option 'savenumberlist=true' required.}%
1433     {%
1434       You must use the 'savenumberlist' package option
1435       to reference location lists.%
1436     }%
1437   \fi
1438 }
```

\@glsdefaultsort Define command to set default sort.

```
1439 \newcommand*{\@glsdefaultsort}{\@glo@name}
```

\gls@level Register to increment entry levels.

```
1440 \newcount\gls@level
```

@gls@noexpand@field

```
1441 \newcommand{\@gls@noexpand@field}[3]{%
1442   \expandafter\global\expandafter
1443     \let\csname glo@#1@#2\endcsname#3%
1444 }
```

gls@noexpand@fields

```
1445 \newcommand{\@gls@noexpand@fields}[4]{%
1446   \ifcsdef{gls@assign@#3@field}
1447     {%
1448       \ifdefequal{#4}{\@gls@default@value}%
1449       {%
1450         \edef\@gls@value{\expandonce{#1}}%
1451         \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1452       }%
1453     }%
1454     \csuse{gls@assign@#3@field}{#2}{#4}%
1455   }%
1456 }%
1457 {%
1458   \ifdefequal{#4}{\@gls@default@value}%
1459   {%
1460     \edef\@gls@value{\expandonce{#1}}%
1461     \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1462   }%
1463   {%
1464     \@gls@noexpand@field{#2}{#3}{#4}%
1465   }%
1466 }%
1467 }
```

\@gls@expand@field

```
1468 \newcommand{\@gls@expand@field}[3]{%
1469   \expandafter
1470     \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1471 }
```

@gls@expand@fields

```
1472 \newcommand{\@gls@expand@fields}[4]{%
1473   \ifcsdef{gls@assign@#3@field}
1474     {%
1475       \ifdefequal{#4}{\@gls@default@value}%
1476       {%
1477         \edef\@gls@value{\expandonce{#1}}%
1478         \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1479       }%
1480     }%
1481     \expandafter\@gls@startswitexpandonce#4\relax\relax\gls@endcheck
```

```

1482      {%
1483      \@@gls@expand@field{#2}{#3}{#4}%
1484      }%
1485      {%
1486      \csuse{gls@assign@#3@field}{#2}{#4}%
1487      }%
1488      }%
1489      }%
1490      {%
1491      \ifdefequal{#4}{\@gls@default@value}%
1492      {%
1493      \@@gls@expand@field{#2}{#3}{#1}%
1494      }%
1495      {%
1496      \@@gls@expand@field{#2}{#3}{#4}%
1497      }%
1498      }%
1499      }

```

startswithexpandonce

```

1500 \def\@gls@expandonce{\expandonce}
1501 \def\@gls@startswithexpandonce#1#2\gls@endcheck#3#4{%
1502   \def\@gls@tmp{#1}%
1503   \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%
1504 }

```

`\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.

```
1505 \let\gls@assign@field\@gls@expand@fields
```

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

```

1506 \newcommand*\glsexpandfields{%
1507   \let\gls@assign@field\@gls@expand@fields
1508 }

```

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

```

1509 \newcommand*\glsnoexpandfields{%
1510   \let\gls@assign@field\@gls@noexpand@fields
1511 }

```

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


```
1512 \newrobustcmd{\newglossaryentry}[2]{%
```

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

```
1513   \glsdoifnoexists{#1}%
1514   {%
1515     \gls@defglossaryentry{#1}{#2}%
1516   }%
1517 }
```

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

```
1518 \newrobustcmd{\provideglossaryentry}[2]{%
1519   \ifglsentryexists{#1}%
1520   }{%
1521   {%
1522     \gls@defglossaryentry{#1}{#2}%
1523   }%
1524 }
1525 \@onlypreamble{\provideglossaryentry}
```

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

```
1526 \newrobustcmd{\new@glossaryentry}[2]{%
1527   \ifundef\@gls@deffile
1528   {%
1529     \global\newwrite\@gls@deffile
1530     \immediate\openout\@gls@deffile=\jobname.glsdefs
1531   }%
1532   {%
1533     \ifglsentryexists{#1}{}%
1534     {%
1535       \gls@defglossaryentry{#1}{#2}%
1536     }%
1537     \@gls@writedef{#1}%
1538   }
1539 \AtBeginDocument
1540 {
1541   \makeatletter
1542   \InputIfFileExists{\jobname.glsdefs}{}{}%
1543   \makeatother
1544   \let\newglossaryentry\new@glossaryentry
1545 }
1546 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}{}%
1547 %   \end{macrocode}
1548 %\end{macro}
1549 %
1550 %\begin{macro}{\@gls@writedef}
1551 %\changes{3.10a}{2013-10-13}{new}
1552 % Writes glossary entry definition to \cs{@gls@deffile}.
1553 %   \begin{macrocode}
```

```

1554 \newcommand*{\@gls@writedef}[1]{%
1555   \immediate\write\@gls@deffile
1556   {%
1557     \string\ifglsentryexists{#1}\}\expandafter\@gobble\string\%^~J%
1558     \expandafter\@gobble\string\{\}\expandafter\@gobble\string\%^~J%
1559     \string\gls@defglossaryentry{#1}\expandafter\@gobble\string\%^~J%
1560     \expandafter\@gobble\string\{\}\expandafter\@gobble\string\%%
1561   }%

  Write key value information:

1562   \@for\@gls@map:=\@gls@keymap\do
1563   {%
1564     \edef\glo@value{\expandafter\expandonce
1565       \csname glo@#1\expandafter\@secondoftwo\@gls@map\endcsname}%
1566     \@onelevel@sanitize\glo@value
1567     \immediate\write\@gls@deffile
1568     {%
1569       \expandafter\@firstoftwo\@gls@map
1570       =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1571       \expandafter\@gobble\string\%%
1572     }%
1573   }%

  Provide hook:

1574   \gls@writedefhook
1575   \immediate\write\@gls@deffile
1576   {%
1577     \expandafter\@gobble\string\%^~J%
1578     \expandafter\@gobble\string\}\expandafter\@gobble\string\%^~J%
1579     \expandafter\@gobble\string\}\expandafter\@gobble\string\%%
1580   }%
1581 }

```

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

```

1582 \newcommand*{\@gls@keymap}{%
1583   {name}{name},%
1584   {sort}{sortvalue},% unescaped sort value
1585   {type}{type},%
1586   {first}{first},%
1587   {firstplural}{firstpl},%
1588   {text}{text},%
1589   {plural}{plural},%
1590   {description}{desc},%
1591   {descriptionplural}{descplural},%
1592   {symbol}{symbol},%
1593   {symbolplural}{symbolplural},%
1594   {user1}{useri},%
1595   {user2}{userii},%
1596   {user3}{useriii},%

```

```

1597 {user4}{useriv},%
1598 {user5}{userv},%
1599 {user6}{uservi},%
1600 {long}{long},%
1601 {longplural}{longpl},%
1602 {short}{short},%
1603 {shortplural}{shortpl},%
1604 {counter}{counter},%
1605 {parent}{parent}%
1606 }

```

`\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.

```

1607 \newcommand*{\glsaddkey}{\@ifstar\sglsaddkey\glsaddkey}

```

Starred version switches on expansion for this key.

```

1608 \newcommand*{\sglsaddkey}[1]{%
1609   \key@ifundefined{glossentry}{#1}%
1610   {%
1611     \expandafter\newcommand\expandafter*\expandafter
1612       {\csname gls@assign@#1@field\endcsname}[2]{%
1613         \@gls@expand@field{##1}{#1}{##2}%
1614       }%
1615   }%
1616   }%
1617   \@glsaddkey{#1}%
1618 }

```

Unstarred version doesn't override default expansion.

```

1619 \newcommand*{\glsaddkey}[7]{%

```

Check the specified key doesn't already exist.

```

1620   \key@ifundefined{glossentry}{#1}%
1621   {%

```

Set up the key.

```

1622     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1623     \appto\@gls@keymap{,{#1}{#1}}%

```

Set the default value.

```

1624     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%

```

Assignment code.

```

1625     \appto\@newglossaryentryposthook{%
1626       \letcs{\@glo@tmp}{@glo@#1}%
1627       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1628     }%

```

Define the no-link commands.

```

1629 \newcommand*{#3}[1]{\csuse{glo@##1@#1}}%
1630 \newcommand*{#4}[1]{%
1631   \letcs{\@glo@text}{glo@##1@#1}%
1632   \xmakefirstuc\@glo@text
1633 }%

```

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

```

1634 \ifcsdef{@gls@user@#1@}%
1635 {%
1636   \PackageError{glossaries}%
1637   {Can't define '\string#5' as helper command
1638   '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1639 }%
1640 }%
1641 {%
1642   \newrobustcmd*{#5}{\@ifstar{\csuse{@sgls@user@#1}}{\csuse{@gls@user@#1}}}%
1643   \expandafter\newcommand\expandafter*\expandafter
1644   {\csname @sgls@user@#1\endcsname}[1][{%
1645     \csuse{@gls@user@#1}[hyper=false,##1]%
1646   }%
1647   \expandafter\newcommand\expandafter*\expandafter
1648   {\csname @gls@user@#1\endcsname}[2][{%
1649     \new@ifnextchar[%
1650       {\csuse{@gls@user@#1@}{##1}{##2}}%
1651       {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
1652   \csdef{@gls@user@#1@}##1##2[##3]{%
1653     \glsdoifexists{##2}%
1654     {%
1655       \edef\@glo@type{\glsentrytype{##2}}%
1656       \@gls@link[##1]{##2}{#3{##2}##3}%
1657     }%
1658   }%
1659 }%

```

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

```

1660 \ifcsdef{@Gls@user@#1@}%
1661 {%
1662   \PackageError{glossaries}%
1663   {Can't define '\string#6' as helper command
1664   '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1665 }%
1666 }%
1667 {%
1668   \newrobustcmd*{#6}{\@ifstar{\csuse{@sGls@user@#1}}{\csuse{@Gls@user@#1}}}%
1669   \expandafter\newcommand\expandafter*\expandafter
1670   {\csname @sGls@user@#1\endcsname}[1][{%
1671     \csuse{@Gls@user@#1}[hyper=false,##1]%
1672   }%
1673   \expandafter\newcommand\expandafter*\expandafter

```

```

1674         {\csname @GLs@user@#1\endcsname}[2] [] {%
1675         \new@ifnextchar [%
1676         {\csuse{@GLs@user@#1@}{##1}{##2}}}%
1677         {\csuse{@GLs@user@#1@}{##1}{##2} []}}}%
1678     \csdef{@GLs@user@#1@}##1##2[##3] {%
1679     \glsdoifexists{##2}%
1680     {%
1681     \edef\@glo@type{\glsentrytype{##2}}%
1682     \@gls@link[##1]{##2}{#4{##2}##3}%
1683     }%
1684     }%
1685     }%

```

Finally the all caps version:

```

1686     \ifcsdef{@GLS@user@#1@}%
1687     {%
1688     \PackageError{glossaries}%
1689     {Can't define '\string#7' as helper command
1690     '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1691     }%
1692     }%
1693     {%
1694     \newrobustcmd*{#7}{\@ifstar{\csuse{@sGLS@user@#1}}{\csuse{@GLS@user@#1}}}%
1695     \expandafter\newcommand\expandafter*\expandafter
1696     {\csname @sGLS@user@#1\endcsname}[1] [] {%
1697     \csuse{@GLS@user@#1}[hyper=false,##1]%
1698     }%
1699     \expandafter\newcommand\expandafter*\expandafter
1700     {\csname @GLS@user@#1\endcsname}[2] [] {%
1701     \new@ifnextchar [%
1702     {\csuse{@GLS@user@#1@}{##1}{##2}}}%
1703     {\csuse{@GLS@user@#1@}{##1}{##2} []}}}%
1704     \csdef{@GLS@user@#1@}##1##2[##3] {%
1705     \glsdoifexists{##2}%
1706     {%
1707     \edef\@glo@type{\glsentrytype{##2}}%
1708     \@gls@link[##1]{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1709     }%
1710     }%
1711     }%
1712     }%
1713     {%
1714     \PackageError{glossaries}{Key '#1' already exists}{}%
1715     }%
1716 }

```

\glswritedefhook

```

1717 \newcommand*\glswritedefhook{}

```

`\gls@assign@desc`

```
1718 \newcommand*{\gls@assign@desc}[1]{%
1719   \gls@assign@field{#1}{desc}{\@glo@desc}%
1720   \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1721 }
```

`\longnewglossaryentry`

```
1722 \newcommand{\longnewglossaryentry}[3]{%
1723   \glsdoifnoexists{#1}%
1724   {%
1725     \bgroup
1726     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1727     \long\def\@newglossaryentryprehook{%
1728       \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1729       \@org@newglossaryentryprehook
1730     }%
1731     \renewcommand*{\gls@assign@desc}[1]{%
1732       \global\cslet{glo@#1@desc}{\@glo@desc}%
1733       \global\cslet{glo@#1@descplural}{\@glo@desc}%
1734     }
1735     \gls@defglossaryentry{#1}{#2}%
1736   \egroup
1737 }
1738 }
```

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

```
1739 \@onlypreamble{\longnewglossaryentry}
```

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

```
1740 \newcommand{\longprovideglossaryentry}[3]{%
1741   \ifglentryexists{#1}{}%
1742   {\longnewglossaryentry{#1}{#2}{#3}}%
1743 }
1744 \@onlypreamble{\longprovideglossaryentry}
```

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

Defines a new entry without checking if it already exists.

```
1745 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1746   \def\@glo@label{#1}%
```

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

```
1747   \let\glslabel\@glo@label
```

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

```

1748 \let\@glo@name\@glsnoname
1749 \let\@glo@desc\@glsnodesc

1750 \let\@glo@descplural\@gls@default@value
1751 \let\@glo@type\@gls@default@value
1752 \let\@glo@symbol\@gls@default@value

1753 \let\@glo@symbolplural\@gls@default@value
1754 \let\@glo@text\@gls@default@value
1755 \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.)

```

1756 \let\@glo@first\@gls@default@value
1757 \let\@glo@firstplural\@gls@default@value

```

Set the default sort:

```

1758 \let\@glo@sort\@gls@default@value

```

Set the default counter:

```

1759 \let\@glo@counter\@gls@default@value

1760 \def\@glo@see{}%

1761 \def\@glo@parent{}%

1762 \def\@glo@prefix{}%

1763 \def\@glo@useri{}%
1764 \def\@glo@userii{}%
1765 \def\@glo@useriii{}%
1766 \def\@glo@useriv{}%
1767 \def\@glo@userv{}%
1768 \def\@glo@uservi{}%

1769 \def\@glo@short{}%
1770 \def\@glo@shortpl{}%
1771 \def\@glo@long{}%
1772 \def\@glo@longpl{}%

```

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

```

1773 \@newglossaryentryprehook

```

Extract key-val information from third parameter:

```

1774 \setkeys{glossentry}{#2}%

```

Check there is a default glossary.

```
1775 \ifundef\glsdefaulttype
1776 {%
1777   \PackageError{glossaries}%
1778     {No default glossary type (have you used 'nomain'?)}%
1779     {If you use package option 'nomain' you must define
1780      a new glossary before you can define entries}%
1781 }%
1782 {}%
```

Assign type. This must be fully expandable

```
1783 \gls@assign@field{\glsdefaulttype}{#1}{type}{\@glo@type}%
1784 \edef\@glo@type{\glsentrytype{#1}}%
```

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

```
1785 \ifcsundef{glolist@\@glo@type}%
1786 {%
1787   \PackageError{glossaries}%
1788     {Glossary type '@glo@type' has not been defined}%
1789     {You need to define a new glossary type, before making entries
1790      in it}%
1791 }%
1792 {%
1793   \protected@edef\glolist@\csname glolist@\@glo@type\endcsname%
1794   \expandafter\xdef\csname glolist@\@glo@type\endcsname{\glolist@{#1},}%
1795 }%
```

Initialise level to 0.

```
1796 \gls@level=0\relax
```

Has this entry been assigned a parent?

```
1797 \ifx\@glo@parent\empty
```

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

```
1798 \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1799 \else
```

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

```
1800 \ifthenelse{\equal{#1}{\@glo@parent}}%
1801 {%
1802   \PackageError{glossaries}{Entry '#1' can't be its own parent}{}%
1803   \def\@glo@parent{}%
1804   \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1805 }%
1806 {}%
```

Check the parent exists:

```
1807 \ifglentryexists{\@glo@parent}%
1808 {%
```

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

```
1809 \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%
```


Determine level.

```
1810      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1811      \advance\gls@level by 1\relax
```

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

```
1812      \ifx\@glo@name\@glsnoname
1813      \expandafter\let\expandafter\@glo@name
1814      \csname glo@\@glo@parent @name\endcsname
```

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

```
1815      \ifx\@glo@plural\@gls@default@value
1816      \expandafter\let\expandafter\@glo@plural
1817      \csname glo@\@glo@parent @plural\endcsname
1818      \fi
1819      \fi
1820      }%
1821      {%
```

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

```
1822      \PackageError{glossaries}%
1823      {%
1824      Invalid parent '\@glo@parent'
1825      for entry '#1' - parent doesn't exist%
1826      }%
1827      {%
1828      Parent entries must be defined before their children%
1829      }%
1830      \def\@glo@parent{}%
1831      \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1832      }%
1833      }%
1834      \fi
```

Set the level for this entry

```
1835      \expandafter\xdef\csname glo@#1@level\endcsname{\number\gls@level}%
```

Define commands associated with this entry:

```
1836      \gls@assign@field{\@glo@name}{#1}{sortvalue}{\@glo@sort}%
1837      \letcs\@glo@sort{glo@#1@sortvalue}%
1838      \gls@assign@field{\@glo@name}{#1}{text}{\@glo@text}%
1839      \expandafter\gls@assign@field\expandafter
1840      {\csname glo@#1@text\endcsname\glspluralsuffix}%
1841      {#1}{plural}{\@glo@plural}%
1842      \expandafter\gls@assign@field\expandafter
1843      {\csname glo@#1@text\endcsname}%
1844      {#1}{first}{\@glo@first}%
```

If first has been specified, make the default by appending \glspluralsuffix, otherwise make the default the value of the plural key.

```
1845      \ifx\@glo@first\@gls@default@value
```

```

1846 \expandafter\gls@assign@field\expandafter
1847 {\csname glo@#1@plural\endcsname}%
1848 {#1}{firstpl}{\@glo@firstplural}%
1849 \else
1850 \expandafter\gls@assign@field\expandafter
1851 {\csname glo@#1@first\endcsname\glspluralsuffix}%
1852 {#1}{firstpl}{\@glo@firstplural}%
1853 \fi

1854 \ifcsundef{@glo@type@\@glo@type @counter}%
1855 {%
1856 \def\@glo@defaultcounter{\glscounter}%
1857 }%
1858 {%
1859 \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
1860 }%
1861 \gls@assign@field{\@glo@defaultcounter}{#1}{counter}{\@glo@counter}%
1862 \gls@assign@field{}{#1}{useri}{\@glo@useri}%
1863 \gls@assign@field{}{#1}{userii}{\@glo@userii}%
1864 \gls@assign@field{}{#1}{useriii}{\@glo@useriii}%
1865 \gls@assign@field{}{#1}{useriv}{\@glo@useriv}%
1866 \gls@assign@field{}{#1}{userv}{\@glo@userv}%
1867 \gls@assign@field{}{#1}{uservi}{\@glo@uservi}%
1868 \gls@assign@field{}{#1}{short}{\@glo@short}%
1869 \gls@assign@field{}{#1}{shortpl}{\@glo@shortpl}%
1870 \gls@assign@field{}{#1}{long}{\@glo@long}%
1871 \gls@assign@field{}{#1}{longpl}{\@glo@longpl}%
1872 \ifx\@glo@name\@glsnname
1873 \@glsnname
1874 \let\@glo@name\@gls@default@value
1875 \fi
1876 \gls@assign@field{}{#1}{name}{\@glo@name}%

```

Set default numberlist if not defined:

```

1877 \ifcsundef{glo@#1@numberlist}%
1878 {%
1879 \csxdef{glo@#1@numberlist}{\noexpand\@gls@missingnumberlist{\@glo@label}}%
1880 }%
1881 {}%

```

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.

```

1882 \def\@glo@@desc{\@glo@first}%
1883 \ifx\@glo@desc\@glo@@desc
1884 \let\@glo@desc\@glo@first
1885 \fi
1886 \ifx\@glo@desc\@glsnodesc
1887 \@glsnodesc
1888 \let\@glo@desc\@gls@default@value

```

```

1889 \fi
1890 \gls@assign@desc{#1}%

```

Set the sort key for this entry:

```

1891 \@gls@defsort{\@glo@type}{#1}%

1892 \def\@glo@@symbol{\@glo@text}%
1893 \ifx\@glo@symbol\@glo@@symbol
1894 \let\@glo@symbol\@glo@text
1895 \fi
1896 \gls@assign@field{\relax}{#1}{symbol}{\@glo@symbol}%
1897 \expandafter
1898 \gls@assign@field\expandafter
1899 {\csname glo@#1@symbol\endcsname}
1900 {#1}{symbolplural}{\@glo@symbolplural}%

```

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

```

1901 \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1902 \expandafter\global\expandafter
1903 \let\csname ifglo@#1@flag\endcsname\iffalse
1904 }%
1905 \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1906 \expandafter\global\expandafter
1907 \let\csname ifglo@#1@flag\endcsname\iftrue
1908 }%
1909 \csname glo@#1@flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1910 \ifx\@glo@see\@empty
1911 \else
1912 \protected@edef\@do@glsee{%
1913 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1914 \noexpand\@nil
1915 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{#1}}%
1916 \@do@glsee
1917 \fi

```

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

```

1918 \do@glo@storeentry{#1}%

```

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

```

1919 \@newglossaryentryposthook
1920 }

```

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

```

1921 \newcommand*{\@newglossaryentryprehook}{}

```

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

```

1922 \newcommand*{\@newglossaryentryposthook}{}

```

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

```

1923 \newcommand*{\glsmoveentry}[2]{%
1924   \edef\glo@type{\csname glo@#1@type\endcsname}%
1925   \def\glo@list{,}%
1926   \forlslentries[\glo@type]{\glo@label}%
1927     {%
1928       \ifthenelse{\equal{\glo@label}{#1}}{\eappto\glo@list{\glo@label,}}%
1929     }%
1930   \cslet\glo@list@\glo@type{\glo@list}%
1931   \csdef{glo@#1@type}{#2}%
1932 }
```

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

```

1933 \ifglxindy
1934   \newcommand*{@glossaryentryfield}{\string\glossentry}
1935 \else
1936   \newcommand*{@glossaryentryfield}{\string\glossentry}
1937 \fi
```

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

```

1938 \ifglxindy
1939   \newcommand*{@glossarysubentryfield}{%
1940     \string\subglossentry}
1941 \else
1942   \newcommand*{@glossarysubentryfield}{%
1943     \string\subglossentry}
1944 \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. The result is stored in `\glo@<label>@entry`, where `<label>` is the entry's label. (This doesn't include any formatting or location information.)

```

1945 \newcommand{\@glo@storeentry}[1]{%
  Escape special characters in the label:
1946   \def\@glo@label{#1}%
1947   \@gls@checkmkidxchars\@glo@label
  Get the sort string and escape any special characters
1948   \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
1949   \@gls@checkmkidxchars\@glo@sort
```

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

```

1950 \@gls@checkmkidxchars\@glo@prefix

Get the parent, if one exists
1951 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%

Write the information to the glossary file.
1952 \ifglxsindy

Store using xindy syntax.
1953 \ifx\@glo@parent\@empty

Entry doesn't have a parent
1954 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1955 (\string"\@glo@sort\string" %
1956 \string"\@glo@prefix\@glossaryentryfield{\@glo@label}\string") %
1957 }%
1958 \else

Entry has a parent
1959 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1960 \csname glo@\@glo@parent @index\endcsname
1961 (\string"\@glo@sort\string" %
1962 \string"\@glo@prefix\@glossarysubentryfield
1963 {\csname glo@#1@level\endcsname}{\@glo@label}\string") %
1964 }%
1965 \fi
1966 \else

Store using makeindex syntax.
1967 \ifx\@glo@parent\@empty

Sanitize \@glo@prefix
1968 \@onelevel@sanitize\@glo@prefix

Entry doesn't have a parent
1969 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1970 \@glo@sort\@gls@actualchar\@glo@prefix
1971 \@glossaryentryfield{\@glo@label}%
1972 }%
1973 \else

Entry has a parent
1974 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1975 \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
1976 \@glo@sort\@gls@actualchar\@glo@prefix
1977 \@glossarysubentryfield
1978 {\csname glo@#1@level\endcsname}{\@glo@label}%
1979 }%
1980 \fi
1981 \fi
1982 }

```

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`

```
1983 \AtBeginDocument{%
1984   \ifpackageloaded{amsmath}%
1985   {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
1986   }%
1987 }
1988 \newcommand*{\@gls@ifnotmeasuring}[1]{%
1989   \ifmeasuring@
1990   \else
1991     #1%
1992   \fi
1993 }
1994 \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.

```
1995 \newcommand*{\glsreset}[1]{%
1996   \gls@ifnotmeasuring
1997   {%
1998     \glsdoifexists{#1}%
1999     {%
2000       \expandafter\global\csname glo@#1@flagfalse\endcsname
2001     }%
2002   }%
2003 }
```

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

```
2004 \newcommand*{\glslocalreset}[1]{%
2005   \gls@ifnotmeasuring
2006   {%
2007     \glsdoifexists{#1}%
2008     {%
2009       \expandafter\let\csname ifglo@#1@flag\endcsname\iffalse
2010     }%
2011   }%
2012 }
```

`\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.

```
2013 \newcommand*{\glsunset}[1]{%
```

```

2014 \gls@ifnotmeasuring
2015 {%
2016     \glsdoifexists{#1}%
2017     {%
2018         \expandafter\global\csname glo@#1@flagtrue\endcsname
2019     }%
2020 }%
2021 }

```

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

```

2022 \newcommand*\glslocalunset}[1]{%
2023     \gls@ifnotmeasuring
2024     {%
2025         \glsdoifexists{#1}%
2026         {%
2027             \expandafter\let\csname ifglo@#1@flag\endcsname\iftrue
2028         }%
2029     }%
2030 }

```

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

Syntax: `\glsresetall[⟨glossary-list⟩]`

`\glsresetall`

```

2031 \newcommand*\glsresetall}[1][\@glo@types]{%
2032     \forallglsentries[#1]{\@glsentry}%
2033     {%
2034         \glsreset{\@glsentry}%
2035     }%
2036 }

```

As above, but with only a local effect:

`\glslocalresetall`

```

2037 \newcommand*\glslocalresetall}[1][\@glo@types]{%
2038     \forallglsentries[#1]{\@glsentry}%
2039     {%
2040         \glslocalreset{\@glsentry}%
2041     }%
2042 }

```

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

Syntax: `\glsunsetall[⟨glossary-list⟩]`

`\glsunsetall`

```

2043 \newcommand*\glsunsetall}[1][\@glo@types]{%
2044     \forallglsentries[#1]{\@glsentry}%
2045     {%
2046         \glsunset{\@glsentry}%
2047     }%
2048 }

```

As above, but with only a local effect:

```
\glslocalunsetall
```

```
2049 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
2050   \forallglsentries[#1]{\@glsentry}%
2051   {%
2052     \glslocalunset{\@glsentry}%
2053   }%
2054 }
```

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
```

```
2055 \newcommand*{\loadglsentries}[2][\@gls@default]{%
2056   \let\@gls@default\glsdefaulttype
2057   \def\glsdefaulttype{#1}\input{#2}%
2058   \let\glsdefaulttype\@gls@default
2059 }
```

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

```
2060 \@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 `\glsformat`. By default this just displays the link text “as is”.

```
\glsformat
```

```
2061 \newcommand*{\glsformat}[1]{#1}
```

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

`\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

```
2062 \newcommand*{\glsentryfmt}{%
2063   \@gls@default@entryfmt\glsdisplayfirst\glsdisplay
2064 }
```

Format that provides backwards compatibility:

```
2065 \newcommand*{\@gls@default@entryfmt}[2]{%
2066   \ifdefempty\glscustomtext
2067     {%
2068       \glsifplural
2069       {%
```

Plural form

```
2070       \glscapscase
2071       {%
```

Don't adjust case

```
2072       \ifglsused\glslabel
2073       {%
```

Subsequent use

```
2074         #2{\glsentryplural{\glslabel}}%
2075         {\glsentrydescplural{\glslabel}}%
2076         {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2077       }%
2078     {%
```

First use

```
2079         #1{\glsentryfirstplural{\glslabel}}%
2080         {\glsentrydescplural{\glslabel}}%
2081         {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2082       }%
2083     }%
2084     {%
```

Make first letter upper case

```
2085       \ifglsused\glslabel
2086       {%
```

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

```
2087       \ifbool{glscompatible-3.07}%
2088       {%
2089         \protected@edef\@glo@etext{%
2090           #2{\glsentryplural{\glslabel}}%
2091           {\glsentrydescplural{\glslabel}}%
```

```

2092         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2093     \xmakefirstuc\@glo@etext
2094 }%
2095 {%
2096     #2{\Glsentryplural{\glslabel}}%
2097     {\glsentrydescplural{\glslabel}}%
2098     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2099 }%
2100 }%
2101 {%

```

First use

```

2102     \ifbool{glscompatible-3.07}%
2103     {%
2104         \protected@edef\@glo@etext{%
2105             #1{\glsentryfirstplural{\glslabel}}%
2106             {\glsentrydescplural{\glslabel}}%
2107             {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2108         \xmakefirstuc\@glo@etext
2109     }%
2110     {%
2111         #1{\Glsentryfirstplural{\glslabel}}%
2112         {\glsentrydescplural{\glslabel}}%
2113         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2114     }%
2115 }%
2116 }%
2117 {%

```

Make all upper case

```

2118     \ifglsused\glslabel
2119     {%

```

Subsequent use

```

2120         \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}%
2121         {\glsentrydescplural{\glslabel}}%
2122         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2123     }%
2124     {%

```

First use

```

2125         \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}%
2126         {\glsentrydescplural{\glslabel}}%
2127         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2128     }%
2129 }%
2130 }%
2131 {%

```

Singular form

```

2132     \glscapscase
2133     {%

```

Don't adjust case

```
2134      \ifglsused\glslabel
2135      {%
```

Subsequent use

```
2136      #2{\glsentrytext{\glslabel}}%
2137      {\glsentrydesc{\glslabel}}%
2138      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2139      }%
2140      {%
```

First use

```
2141      #1{\glsentryfirst{\glslabel}}%
2142      {\glsentrydesc{\glslabel}}%
2143      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2144      }%
2145      }%
2146      {%
```

Make first letter upper case

```
2147      \ifglsused\glslabel
2148      {%
```

Subsequent use

```
2149      \ifbool{glscompatible-3.07}%
2150      {%
2151      \protected@edef\@glo@etext{%
2152      #2{\glsentrytext{\glslabel}}%
2153      {\glsentrydesc{\glslabel}}%
2154      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2155      \xmakefirstuc\@glo@etext
2156      }%
2157      {%
2158      #2{\Glsentrytext{\glslabel}}%
2159      {\glsentrydesc{\glslabel}}%
2160      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2161      }%
2162      }%
2163      {%
```

First use

```
2164      \ifbool{glscompatible-3.07}%
2165      {%
2166      \protected@edef\@glo@etext{%
2167      #1{\glsentryfirst{\glslabel}}%
2168      {\glsentrydesc{\glslabel}}%
2169      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2170      \xmakefirstuc\@glo@etext
2171      }%
2172      {%
2173      #1{\Glsentryfirst{\glslabel}}%
```

```

2174         {\glsentrydesc{\glslabel}}}%
2175         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2176     }%
2177 }%
2178 }%
2179 {%

    Make all upper case
2180     \ifglsused\glslabel
2181     {%

        Subsequent use
2182         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}}%
2183         {\glsentrydesc{\glslabel}}}%
2184         {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2185     }%
2186     {%

        First use
2187         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}}%
2188         {\glsentrydesc{\glslabel}}}%
2189         {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2190     }%
2191 }%
2192 }%
2193 }%
2194 {%

    Custom text provided in \glsdisp
2195     \ifglsused{\glslabel}}%
2196     {%

        Subsequent use
2197         #2{\glscustomtext}}%
2198         {\glsentrydesc{\glslabel}}}%
2199         {\glsentrysymbol{\glslabel}}{\}%
2200     }%
2201     {%

        First use
2202         #1{\glscustomtext}}%
2203         {\glsentrydesc{\glslabel}}}%
2204         {\glsentrysymbol{\glslabel}}{\}%
2205     }%
2206 }%
2207 }

```

`\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.

```

2208 \newcommand*{\glsgenentryfmt}{%
2209     \ifdefempty\glscustomtext

```

```

2210  {%
2211    \glsifplural
2212  {%

  Plural form
2213    \glscapscase
2214  {%

  Don't adjust case
2215    \ifglsused\glslabel
2216  {%

  Subsequent use
2217    \glsentryplural{\glslabel}\glsinsert
2218  }%
2219  {%

  First use
2220    \glsentryfirstplural{\glslabel}\glsinsert
2221  }%
2222  }%
2223  {%

  Make first letter upper case
2224    \ifglsused\glslabel
2225  {%

  Subsequent use.
2226    \Glsentryplural{\glslabel}\glsinsert
2227  }%
2228  {%

  First use
2229    \Glsentryfirstplural{\glslabel}\glsinsert
2230  }%
2231  }%
2232  {%

  Make all upper case
2233    \ifglsused\glslabel
2234  {%

  Subsequent use
2235    \mfirstucMakeUppercase
2236    {\glsentryplural{\glslabel}\glsinsert}%
2237  }%
2238  {%

  First use
2239    \mfirstucMakeUppercase
2240    {\glsentryfirstplural{\glslabel}\glsinsert}%
2241  }%
2242  }%

```

2243 }%
 2244 {%

Singular form

2245 \glscapscase
 2246 {%

Don't adjust case

2247 \ifglused\glslabel
 2248 {%

Subsequent use

2249 \glentrytext{\glslabel}\glinsert
 2250 }%
 2251 {%

First use

2252 \glentryfirst{\glslabel}\glinsert
 2253 }%
 2254 }%
 2255 {%

Make first letter upper case

2256 \ifglused\glslabel
 2257 {%

Subsequent use

2258 \Glentrytext{\glslabel}\glinsert
 2259 }%
 2260 {%

First use

2261 \Glentryfirst{\glslabel}\glinsert
 2262 }%
 2263 }%
 2264 {%

Make all upper case

2265 \ifglused\glslabel
 2266 {%

Subsequent use

2267 \mfirstucMakeUppercase{\glentrytext{\glslabel}\glinsert}%
 2268 }%
 2269 {%

First use

2270 \mfirstucMakeUppercase{\glentryfirst{\glslabel}\glinsert}%
 2271 }%
 2272 }%
 2273 }%
 2274 }%
 2275 {%

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

```
2276 \glscustomtext\glsinsert
2277 }%
2278 }
```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2279 \newcommand*{\glsdisplayfirst}[4]{#1#4}
```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2280 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2281 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2282 \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2283 Use \string\defglsentryfmt\space instead}%
2284 \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2285 \edef\@gls@doentrydef{%
2286 \noexpand\defglsentryfmt[#1]{%
2287 \noexpand\ifcsdef{gls@#1@displayfirst}%
2288 {%
2289 \noexpand\@gls@default@entryfmt
2290 {\noexpand\csuse{gls@#1@displayfirst}}
2291 {\noexpand\csuse{gls@#1@display}}}%
2292 }%
2293 {%
2294 \noexpand\@gls@default@entryfmt
2295 {\noexpand\glsdisplayfirst}
2296 {\noexpand\csuse{gls@#1@display}}}%
2297 }%
2298 }%
2299 }%
2300 \@gls@doentrydef
2301 }
```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2302 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2303 \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2304 Use \string\defglsentryfmt\space instead}%
2305 \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2306 \edef\@gls@doentrydef{%
2307 \noexpand\defglsentryfmt[#1]{%
2308 \noexpand\ifcsdef{gls@#1@display}%
2309 {%
2310 \noexpand\@gls@default@entryfmt
2311 {\noexpand\csuse{gls@#1@displayfirst}}
2312 {\noexpand\csuse{gls@#1@display}}}%
2313 }%
2314 }
```

```

2314      {%
2315      \noexpand\@gls@default@entryfmt
2316      {\noexpand\csuse{gls@#1@displayfirst}}}%
2317      {\noexpand\glsdisplay}
2318      }%
2319  }%
2320 }%
2321 \@gls@doentrydef
2322 }

```

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.

```

2323 \define@key{glslink}{counter}{%
2324   \ifcsundef{c@#1}%
2325   {%
2326     \PackageError{glossaries}%
2327     {There is no counter called ‘#1’}%
2328     {%
2329       The counter key should have the name of a valid counter
2330       as its value%
2331     }%
2332   }%
2333   {%
2334     \def\@gls@counter{#1}%
2335   }%
2336 }

```

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.

```

2337 \define@key{glslink}{format}{%
2338 \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.

```
2339 \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.

```
2340 \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
```

```
2341 \newrobustcmd*{\glslink}{%
2342 \@ifstar\@sgls@link\@gls@link
2343 }
```

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

```
2344 \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.

```
2345 \newcommand*{\@gls@link}[3][\@gls@link]{%
2346 \ifglsentryexists{#2}%
2347 {%
2348 \@gls@link[#1]{#2}{#3}%
2349 }{%
2350 \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2351 defined}{You need to define a glossary entry before you
2352 can use it.}%

```

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

```
2353 \glstextformat{#3}%
2354 }%
2355 }
```

`\@gls@link`

```
2356 \def\@gls@link[#1]#2#3{%
```

Inserting `\leavevmode` suggested by Donald Arseneau (avoids problem with `tabularx`).

```
2357 \leavevmode
```

```
2358 \def\glslabel{#2}%
```

```
2359 \def\@glsnumberformat{glsnumberformat}%
```

```
2360 \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
```

If this is in one of the “nohypertypes” glossaries, suppress the hyperlink by default

```
2361 \edef\gls@type{\csname glo@#2@type\endcsname}%
```

```
2362 \expandafter\DTLifinlist\expandafter
```

```
2363 {\gls@type}{\@gls@nohyperlist}%
```

```
2364 {%
```

```
2365 \KV@glslink@hyperfalse
```

```
2366 }%
```

```
2367 {%
```

```
2368 \KV@glslink@hypertrue
```

```
2369 }%
```

```
2370 \setkeys{glslink}{#1}%
```

Store the entry’s counter in `\theglsentrycounter`

```
2371 \@gls@saveentrycounter
```

Define sort key if necessary:

```
2372 \@gls@setsort{#2}%
```

```
2373 \@do@wrglossary{#2}%
```

```
2374 \ifKV@glslink@hyper
```

```
2375 \@glslink{\glo@linkprefix#2}{\glstextformat{#3}}%
```

```
2376 \else
```

```
2377 \glstextformat{#3}%
```

```
2378 \fi
```

```
2379 }
```

`\glo@linkprefix`

```
2380 \newcommand*{\glo@linkprefix}{glo:}
```

`\glsentrycounter` Set default value of entry counter

```
2381 \def\glsentrycounter{\glscounter}%
```

`\@gls@saveentrycounter` Need to check if using equation counter in align environment:

```
2382 \newcommand*{\@gls@saveentrycounter}{%
```

```
2383 \def\@gls@Hcounter{}}%
```

Are we using equation counter?

```
2384 \ifthenelse{\equal{\@gls@counter}{equation}}{%
```

```
2385 {
```

If we in align environment, \xatlevel@ will be defined. (Can't test for \@currentvir as may be inside an inner environment.)

```

2386 \ifcsundef{xatlevel@}%
2387 {%
2388 \edef\theglsentrycounter{\expandafter\noexpand
2389 \csname the\@gls@counter\endcsname}%
2390 }%
2391 {%
2392 \ifx\xatlevel@\@empty
2393 \edef\theglsentrycounter{\expandafter\noexpand
2394 \csname the\@gls@counter\endcsname}%
2395 \else
2396 \savecounters@
2397 \advance\c@equation by 1\relax
2398 \edef\theglsentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

2399 \ifcsundef{theH\@gls@counter}%
2400 {%
2401 \def\@gls@Hcounter{\theglsentrycounter}%
2402 }%
2403 {%
2404 \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2405 }%
2406 \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2407 \restorecounters@
2408 \fi
2409 }%
2410 }%
2411 {%

```

Not using equation counter so no special measures:

```

2412 \edef\theglsentrycounter{\expandafter\noexpand
2413 \csname the\@gls@counter\endcsname}%
2414 }%

```

Check if hyperref version of this counter

```

2415 \ifx\@gls@Hcounter\@empty
2416 \ifcsundef{theH\@gls@counter}%
2417 {%
2418 \def\theHglentrycounter{\theglsentrycounter}%
2419 }%
2420 {%
2421 \protected@edef\theHglentrycounter{\expandafter\noexpand
2422 \csname theH\@gls@counter\endcsname}%
2423 }%
2424 \fi
2425 }

```

`\@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.

```

2426 \def\@set@glo@numformat#1#2#3#4{%
2427   \expandafter\@glo@check@mkidxrangechar#3\@nil
2428   \protected@edef#1{%
2429     \@glo@prefix setentrycounter[#4]{#2}%
2430     \expandafter\string\csname\@glo@suffix\endcsname
2431   }%
2432   \@gls@checkmkidxchars#1%
2433 }

```

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.

```

2434 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2435   \if#1(\relax
2436     \def\@glo@prefix{(%}
2437     \if\relax#2\relax
2438       \def\@glo@suffix{glsnumberformat}%
2439     \else
2440       \def\@glo@suffix{#2}%
2441     \fi
2442   \else
2443     \if#1)\relax
2444       \def\@glo@prefix{)%}
2445     \if\relax#2\relax
2446       \def\@glo@suffix{glsnumberformat}%
2447     \else
2448       \def\@glo@suffix{#2}%
2449     \fi
2450   \else
2451     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2452   \fi
2453 }

```

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

```

2454 \newcommand*\@gls@escbsdq[1]{%
2455   \def\@gls@checkedmkidx{%}
2456   \let\gls@xdystring=#1\relax
2457   \@onelevel@sanitize\gls@xdystring
2458   \edef\do@gls@xdycheckbackslash{%
2459     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2460     \@backslashchar\@backslashchar\noexpand\null}%

```

```

2461 \do@gl@xdycheckbackslash
2462 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%
2463 \def\@gl@checkedmkidx{%
2464 \expandafter\@gl@xdycheckquote\gl@xdystring\@nil""\null
2465 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%

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

2466 \@for\@gl@tmp:=\gl@protected@pagefmts\do
2467 {%
2468 \edef\@gl@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gl@tmp}%
2469 \@onelevel@sanitize\@gl@sanitized@tmp
2470 \edef\gl@dostubst{%
2471 \noexpand\DTLsubstituteall\noexpand\gl@xdystring
2472 {\@gl@sanitized@tmp}{\expandonce\@gl@tmp}%
2473 }%
2474 \gl@dostubst
2475 }%

Assign to required control sequence
2476 \let#1=\gl@xdystring
2477 }

```

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

gl@checkmkidxchars

```

2478 \newcommand{\@gl@checkmkidxchars}[1]{%
2479 \ifgl@xindy
2480 \@gl@escbsdq{#1}%
2481 \else
2482 \def\@gl@checkedmkidx{%
2483 \expandafter\@gl@checkquote#1\@nil""\null
2484 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2485 \def\@gl@checkedmkidx{%
2486 \expandafter\@gl@checkescquote#1\@nil\""\null
2487 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2488 \def\@gl@checkedmkidx{%
2489 \expandafter\@gl@checkescactual#1\@nil\?\?\null
2490 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2491 \def\@gl@checkedmkidx{%
2492 \expandafter\@gl@checkactual#1\@nil??\null
2493 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2494 \def\@gl@checkedmkidx{%
2495 \expandafter\@gl@checkbar#1\@nil||\null
2496 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2497 \def\@gl@checkedmkidx{%
2498 \expandafter\@gl@checkescbar#1\@nil\\|\null
2499 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2500 \def\@gl@checkedmkidx{%
2501 \expandafter\@gl@checklevel#1\@nil!!\null

```

```

2502 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2503 \fi
2504 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

2505 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

2506 \newtoks\@gls@tmpb

```

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

```

2507 \def\@gls@checkquote#1"#2"#3\null{%
2508 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2509 \toks@={#1}%
2510 \ifx\null#2\null
2511 \ifx\null#3\null
2512 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2513 \def\@gls@checkquote{\relax}%
2514 \else
2515 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2516 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2517 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2518 \fi
2519 \else
2520 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2521 \@gls@quotechar\@gls@quotechar}%
2522 \ifx\null#3\null
2523 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2524 \else
2525 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2526 \fi
2527 \fi
2528 \@gls@checkquote
2529 }

```

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

```

2530 \def\@gls@checkescquote#1\"#2\"#3\null{%
2531 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2532 \toks@={#1}%
2533 \ifx\null#2\null
2534 \ifx\null#3\null
2535 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2536 \def\@gls@checkescquote{\relax}%
2537 \else
2538 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2539 \@gls@quotechar\string\" \@gls@quotechar
2540 \@gls@quotechar\string\" \@gls@quotechar}%

```

```

2541 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2542 \fi
2543 \else
2544 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2545 \@gls@quotearchar\string"\@gls@quotearchar}%
2546 \ifx\null#3\null
2547 \def\@gls@checkescquote{\@gls@checkescquote#2\""\null}%
2548 \else
2549 \def\@gls@checkescquote{\@gls@checkescquote#2\">#3\null}%
2550 \fi
2551 \fi
2552 \@gls@checkescquote
2553 }

```

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

```

2554 \def\@gls@checkescactual#1\?#2\?#3\null{%
2555 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2556 \toks@={#1}%
2557 \ifx\null#2\null
2558 \ifx\null#3\null
2559 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2560 \def\@gls@checkescactual{\relax}%
2561 \else
2562 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2563 \@gls@quotearchar\string"\@gls@actualchar
2564 \@gls@quotearchar\string"\@gls@actualchar}%
2565 \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
2566 \fi
2567 \else
2568 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2569 \@gls@quotearchar\string"\@gls@actualchar}%
2570 \ifx\null#3\null
2571 \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2572 \else
2573 \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2574 \fi
2575 \fi
2576 \@gls@checkescactual
2577 }

```

`\@gls@checkescbar` Similarly for `\|`:

```

2578 \def\@gls@checkescbar#1\|#2\|#3\null{%
2579 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2580 \toks@={#1}%
2581 \ifx\null#2\null
2582 \ifx\null#3\null
2583 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2584 \def\@gls@checkescbar{\relax}%
2585 \else

```

```

2586 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2587 \@gls@quotearchar\string\"@gls@encapchar
2588 \@gls@quotearchar\string\"@gls@encapchar}%
2589 \def\@@gls@checkescbar{\@gls@checkescbar#3\null}%
2590 \fi
2591 \else
2592 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2593 \@gls@quotearchar\string\"@gls@encapchar}%
2594 \ifx\null#3\null
2595 \def\@@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2596 \else
2597 \def\@@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2598 \fi
2599 \fi
2600 \@@gls@checkescbar
2601 }

```

\@gls@checkesclevel Similarly for \!:

```

2602 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2603 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2604 \toks@={#1}%
2605 \ifx\null#2\null
2606 \ifx\null#3\null
2607 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2608 \def\@@gls@checkesclevel{\relax}%
2609 \else
2610 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2611 \@gls@quotearchar\string\"@gls@levelchar
2612 \@gls@quotearchar\string\"@gls@levelchar}%
2613 \def\@@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2614 \fi
2615 \else
2616 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2617 \@gls@quotearchar\string\"@gls@levelchar}%
2618 \ifx\null#3\null
2619 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2620 \else
2621 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2622 \fi
2623 \fi
2624 \@@gls@checkesclevel
2625 }

```

\@gls@checkbar and for |:

```

2626 \def\@gls@checkbar#1|#2|#3\null{%
2627 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2628 \toks@={#1}%
2629 \ifx\null#2\null
2630 \ifx\null#3\null

```



```

2631 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2632 \def\@@gls@checkbar{\relax}%
2633 \else
2634 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2635 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
2636 \def\@@gls@checkbar{\@gls@checkbar#3\null}%
2637 \fi
2638 \else
2639 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2640 \@gls@quotechar\@gls@encapchar}%
2641 \ifx\null#3\null
2642 \def\@@gls@checkbar{\@gls@checkbar#2||\null}%
2643 \else
2644 \def\@@gls@checkbar{\@gls@checkbar#2|#3\null}%
2645 \fi
2646 \fi
2647 \@@gls@checkbar
2648 }

```

\@gls@checklevel and for !:

```

2649 \def\@gls@checklevel#1!#2!#3\null{%
2650 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2651 \toks@={#1}%
2652 \ifx\null#2\null
2653 \ifx\null#3\null
2654 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2655 \def\@@gls@checklevel{\relax}%
2656 \else
2657 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2658 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
2659 \def\@@gls@checklevel{\@gls@checklevel#3\null}%
2660 \fi
2661 \else
2662 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2663 \@gls@quotechar\@gls@levelchar}%
2664 \ifx\null#3\null
2665 \def\@@gls@checklevel{\@gls@checklevel#2!!\null}%
2666 \else
2667 \def\@@gls@checklevel{\@gls@checklevel#2!#3\null}%
2668 \fi
2669 \fi
2670 \@@gls@checklevel
2671 }

```

\@gls@checkactual and for ?:

```

2672 \def\@gls@checkactual#1?#2?#3\null{%
2673 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2674 \toks@={#1}%
2675 \ifx\null#2\null

```

```

2676 \ifx\null#3\null
2677 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2678 \def\@@gls@checkactual{\relax}%
2679 \else
2680 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2681 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
2682 \def\@@gls@checkactual{\@gls@checkactual#3\null}%
2683 \fi
2684 \else
2685 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2686 \@gls@quotechar\@gls@actualchar}%
2687 \ifx\null#3\null
2688 \def\@@gls@checkactual{\@gls@checkactual#2??\null}%
2689 \else
2690 \def\@@gls@checkactual{\@gls@checkactual#2?#3\null}%
2691 \fi
2692 \fi
2693 \@@gls@checkactual
2694 }

```

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

```

2695 \def\@gls@xdycheckquote#1"#2"#3\null{%
2696 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2697 \toks@={#1}%
2698 \ifx\null#2\null
2699 \ifx\null#3\null
2700 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2701 \def\@@gls@xdycheckquote{\relax}%
2702 \else
2703 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2704 \string\}\string\}%
2705 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
2706 \fi
2707 \else
2708 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2709 \string\}%
2710 \ifx\null#3\null
2711 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
2712 \else
2713 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
2714 \fi
2715 \fi
2716 \@@gls@xdycheckquote
2717 }

```

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

```

\@gls@xdycheckbackslash
2718 \edef\def\@gls@xdycheckbackslash{%
2719 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar

```

```

2720    ##2\@backslashchar##3\noexpand\null{%
2721 \noexpand\@gls@tmpb=\noexpand\expandafter
2722   {\noexpand\@gls@checkedmkidx}%
2723 \noexpand\toks@={##1}%
2724 \noexpand\ifx\noexpand\null##2\noexpand\null
2725 \noexpand\ifx\noexpand\null##3\noexpand\null
2726 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2727   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
2728 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
2729 \noexpand\else
2730 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2731   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2732   \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
2733 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2734   \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
2735 \noexpand\fi
2736 \noexpand\else
2737 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2738   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2739   \@backslashchar\@backslashchar}%
2740 \noexpand\ifx\noexpand\null##3\noexpand\null
2741 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2742   \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2743   \@backslashchar\noexpand\null}%
2744 \noexpand\else
2745 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2746   \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2747   ##3\noexpand\null}%
2748 \noexpand\fi
2749 \noexpand\fi
2750 \noexpand\@gls@xdycheckbackslash
2751 }%
2752 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

2753 \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.

```

2754 \ifcsundef{hyperlink}%
2755 {%
2756   \gdef\@glslink#1#2{#2}%
2757 }%
2758 {%
2759   \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
2760 }

```

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

```

2761 \newlength\gls@tmplen \ifcsundef{hypertarget}%
2762 {%
2763   \gdef\@gls@target#1#2{#2}%
2764 }%
2765 {%
2766   \gdef\@gls@target#1#2{%
2767     \settoheight{\gls@tmplen}{#2}%
2768     \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
2769   }%
2770 }

```

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

`\glsdisablehyper`

```

2771 \newcommand{\glsdisablehyper}{%
2772   \renewcommand*\@glslink[2]{##2}%
2773   \renewcommand*\@gls@target[2]{##2}%
2774 }

```

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

`\glsenablehyper`

```

2775 \newcommand{\glsenablehyper}{%
2776   \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
2777   \renewcommand*\@gls@target[2]{%
2778     \settoheight{\gls@tmplen}{##2}%
2779     \raisebox{\gls@tmplen}{\hypertarget{##1}{}}##2}}

```

Provide some convenience commands if not already defined:

```

2780 \providecommand{\@firstofthree}[3]{#1}
2781 \providecommand{\@secondofthree}[3]{#2}
2782 \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`

```
2783 \newrobustcmd*{\gls}{\@ifstar\@sgls\gls}
```

Define the starred form:

`\@sgls`

```
2784 \newcommand*{\@sgls}[1] [] {\@gls[hyper=false,#1]}
```

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

`\@gls`

```
2785 \newcommand*{\@gls}[2] [] {%
2786   \new@ifnextchar[{\@gls@{#1}{#2}}{\@gls@{#1}{#2} []}%
2787 }
```

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

```
2788 \def\@gls@#1#2[#3] {%
2789   \glsdoifexists{#2}%
2790   {%
2791     \edef\@glo@type{\glsentrytype{#2}}%
2792     \def\@gls@link@opts{#1}%
2793     \def\@gls@link@label{#2}%
2794     \def\glslabel{#2}%
2795     \let\glsifplural\@secondoftwo
2796     \let\glsacscase\@firstofthree
2797     \let\glscustomtext\@empty
2798     \def\glsinsert{#3}%
```

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

```
2799     \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.

```
2800     \ifglsused{#2}%
2801     {%
2802       \@gls@link[#1]{#2}{\@glo@text}%
2803     }%
2804     {%
2805       \gls@checkisacronymlist\@glo@type
2806       \ifthenelse
2807       {(\boolean{glsisacronymlist}\AND \boolean{glsacrfootnote})\OR
2808        \NOT\boolean{glshyperfirst}}
2809       {%
2810       }%
2811       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
```

```

2812     }%
2813     {%
2814     \@gls@link[#1]{#2}{\@glo@text}%
2815     }%
2816     }%

```

Indicate that this entry has now been used

```

2817     \ifKV@glslink@local
2818     \glslocalunset{#2}%
2819     \else
2820     \glsunset{#2}%
2821     \fi
2822     }%
2823 }

```

`\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`

```

2824 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

2825 \newcommand*{\@sGls}[1][\@Gls[hyper=false,#1]]{

```

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

```

2826 \newcommand*{\@Gls}[2][\@Gls@#1]{#2}%
2827 \new@ifnextchar[\@Gls@#1]{#2}{\@Gls@#1}{#2}[]}%
2828 }

```

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

```

2829 \def\@Gls@#1#2[#3]{%
2830   \glsdoifexists{#2}%
2831   {%
2832     \edef\@glo@type{\glsentrytype{#2}}%

```

Save options in `\@gls@link@opts` and label in `\@gls@link@label`

```

2833   \def\@gls@link@opts{#1}%
2834   \def\@gls@link@label{#2}%
2835   \def\glslabel{#2}%

2836   \let\glsifplural\@secondoftwo
2837   \let\glsupcase\@secondofthree
2838   \let\glscustomtext\@empty
2839   \def\glsinsert{#3}%

```

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

```

2840   \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.

```

2841 \ifglsused{#2}%
2842 {%
2843 \@gls@link[#1]{#2}{\@glo@text}%
2844 }%
2845 {%
2846 \gls@checkisacronymlist\@glo@type
2847 \ifthenelse
2848 {%
2849 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2850 \OR \NOT\boolean{glshyperfirst}}%
2851 }%
2852 {%
2853 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2854 }%
2855 {%
2856 \@gls@link[#1]{#2}{\@glo@text}%
2857 }%
2858 }%

```

Indicate that this entry has now been used

```

2859 \ifKV@glslink@local
2860 \glslocalunset{#2}%
2861 \else
2862 \glsunset{#2}%
2863 \fi
2864 }%
2865 }

```

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

`\GLS`

```

2866 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

2867 \newcommand*{\@sGLS}[1][\@GLS[hyper=false,#1]]

```

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

```

2868 \newcommand*{\@GLS}[2][\@GLS@{#1}{#2}]{\@GLS@{#1}{#2}[]}
2869 \new@ifnextchar[\@GLS@{#1}{#2}]{\@GLS@{#1}{#2}[]}
2870 }

```

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

```

2871 \def\@GLS@#1#2[#3]{%
2872 \glsdoifexists{#2}%
2873 {%
2874 \edef\@glo@type{\glsentrytype{#2}}%

```

Save options in \@gls@link@opts and label in \@gls@link@label

```
2875 \def\@gls@link@opts{#1}%
2876 \def\@gls@link@label{#2}%

2877 \def\glslabel{#2}%
2878 \let\glsifplural\@secondoftwo
2879 \let\glscapscase\@thirdofthree
2880 \let\glscustomtext\@empty
2881 \def\glsinsert{#3}%
```

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

```
2882 \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.

```
2883 \ifglsused{#2}%
2884 {%
2885   \@gls@link[#1]{#2}{\@glo@text}%
2886 }%
2887 {%
2888   \gls@checkisacronymlist\@glo@type
2889   \ifthenelse
2890   {%
2891     \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2892     \OR \NOT\boolean{glshyperfirst}}{%
2893     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2894   }%
2895   {%
2896     \@gls@link[#1]{#2}{\@glo@text}%
2897   }%
2898 }%
```

Indicate that this entry has now been used

```
2899 \ifKV@glslink@local
2900   \glslocalunset{#2}%
2901 \else
2902   \glsunset{#2}%
2903 \fi
2904 }%
2905 }
```

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

\glspl

```
2906 \newrobustcmd*{\glspl}{\@ifstar\@sglspl\@glspl}
```

Define the starred form:

```
2907 \newcommand*{\@sglspl}[1][\@glspl[hyper=false,#1]]
```


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

```
2908 \newcommand*{\@glspl}[2][\%
2909 \new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}%
2910 }
```

\@glspl@ Read in the final optional argument:

```
2911 \def\@glspl@#1#2[#3]{%
2912 \glsdoifexists{#2}%
2913 {%
2914 \edef\@glo@type{\glsentrytype{#2}}%
2915 \def\@gls@link@opts{#1}%
2916 \def\@gls@link@label{#2}%
2917 \def\glslabel{#2}%
2918 \let\glsifplural\@firstoftwo
2919 \let\glsapscase\@firstofthree
2920 \let\glscustomtext\@empty
2921 \def\glsinsert{#3}%
2922 % Determine what the link text should be (this is stored in
2923 % \cs{@glo@text})
2924 %\changes{1.12}{2008 Mar 8}{now uses \cs{glsentrydescplural} and
2925 % \cs{glsentrysymbolplural} instead of \cs{glsentrydesc} and
2926 % \cs{glsentrysymbol}}
2927 %\changes{3.11a}{2013-10-15}{change to using \cs{glsentryfmt} style
2928 %commands}
2929 % \begin{macrocode}
2930 \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.

```
2931 \ifglsused{#2}%
2932 {%
2933 \@gls@link[#1]{#2}{\@glo@text}%
2934 }%
2935 {%
2936 \gls@checkisacronymlist\@glo@type
2937 \ifthenelse
2938 {%
2939 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2940 \OR \NOT\boolean{glshyperfirst}}%
2941 }%
2942 {%
2943 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2944 }%
2945 {%

```

```

2946      \@gls@link[#1]{#2}{\@glo@text}%
2947    }%
2948  }%

```

Indicate that this entry has now been used

```

2949    \ifKV@glslink@local
2950      \glslocalunset{#2}%
2951    \else
2952      \glsunset{#2}%
2953    \fi
2954  }%
2955 }

```

`\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`

```

2956 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

2957 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]{

```

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

```

2958 \newcommand*{\@Glspl}[2][\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}
2959 \new@ifnextchar{\@Glspl@{#1}{#2}}{\@Glspl@{#1}{#2}[]}
2960 }

```

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

```

2961 \def\@Glspl@#1#2[#3]{%
2962   \glsdoifexists{#2}%
2963   {%
2964     \edef\@glo@type{\glsentrytype{#2}}%

```

Save options in `\@gls@link@opts` and label in `\@gls@link@label`

```

2965   \def\@gls@link@opts{#1}%
2966   \def\@gls@link@label{#2}%
2967   \def\glslabel{#2}%
2968   \let\glsifplural\@firstoftwo
2969   \let\glscapscase\@secondofthree
2970   \let\glscustomtext\@empty
2971   \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`.

```

2972   \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.

```

2973   \ifglsused{#2}%
2974   {%
2975     \@gls@link[#1]{#2}{\@glo@text}%
2976   }%
2977   {%
2978     \gls@checkisacronymlist\@glo@type
2979     \ifthenelse
2980     {%
2981       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2982       \OR \NOT\boolean{glshyperfirst}}%
2983     }%
2984     {%
2985       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2986     }%
2987     {%
2988       \@gls@link[#1]{#2}{\@glo@text}%
2989     }%
2990   }%

```

Indicate that this entry has now been used

```

2991   \ifKV@glslink@local
2992   \glslocalunset{#2}%
2993   \else
2994   \glsunset{#2}%
2995   \fi
2996 }%
2997 }

```

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

`\GLSp1`

```

2998 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

2999 \newcommand*{\@sGLSp1}[1][\@GLSp1[hyper=false,#1]]

```

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

```

3000 \newcommand*{\@GLSp1}[2][\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}
3001 \new@ifnextchar[\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}
3002 }

```

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

```

3003 \def\@GLSp1@#1#2[#3]{%
3004   \glsdoifexists{#2}%

```

```

3005  {%
3006    \edef\@glo@type{\glsentrytype{#2}}%
      Save options in \@gls@link@opts and label in \@gls@link@label
3007    \def\@gls@link@opts{#1}%
3008    \def\@gls@link@label{#2}%

3009    \def\glslabel{#2}%
3010    \let\glsifplural\@firstoftwo
3011    \let\glscapscase\@thirdofthree
3012    \let\glscustomtext\@empty
3013    \def\glsinsert{#3}%

      Determine what the link text should be (this is stored in \@glo@text)
3014    \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.
3015    \ifglsused{#2}%
3016    {%
3017      \@gls@link[#1]{#2}{\@glo@text}%
3018    }%
3019    {%
3020      \gls@checkisacronymlist\@glo@type
3021      \ifthenelse
3022      {%
3023        \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3024        \OR \NOT\boolean{glshyperfirst}}%
3025      }%
3026      {%
3027        \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3028      }%
3029      {%
3030        \@gls@link[#1]{#2}{\@glo@text}%
3031      }%
3032    }%

      Indicate that this entry has now been used
3033    \ifKV@glslink@local
3034      \glslocalunset{#2}%
3035    \else
3036      \glsunset{#2}%
3037    \fi
3038  }%
3039 }

```

`\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:

```
3040 \newrobustcmd*{\glsdisp}{\@ifstar\sglsdisp\@glsdisp}
```

Define the starred form:

\@sgls

```
3041 \newcommand*{\@sglsdisp}[1] [] {\@glsdisp[hyper=false,#1]}
```

Defined the un-starred form.

\@glsdisp

```
3042 \newcommand*{\@glsdisp}[3] [] {%
```

```
3043   \glsdoifexists{#2}{%
```

```
3044     \edef\@glo@type{\glsentrytype{#2}}%
```

Save options in \@gls@link@opts and label in \@gls@link@label

```
3045   \def\@gls@link@opts{#1}%
```

```
3046   \def\@gls@link@label{#2}%
```

```
3047   \def\glslabel{#2}%
```

```
3048   \let\glsifplural\@secondoftwo
```

```
3049   \let\glsapscase\@firstofthree
```

```
3050   \def\glscustomtext{#3}%
```

```
3051   \def\glsinsert{}%
```

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

```
3052   \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.

```
3053   \ifglsused{#2}%
```

```
3054   {%
```

```
3055     \@gls@link[#1]{#2}{\@glo@text}%
```

```
3056   }%
```

```
3057   {%
```

```
3058     \gls@checkisacronymlist\@glo@type
```

```
3059     \ifthenelse{\boolean{@glsisacronymlist}}{\AND
```

```
3060       \boolean{glsacrfootnote}}{\OR \NOT\boolean{gls hyperfirst}}}%
```

```
3061     {%
```

```
3062       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
```

```
3063     }%
```

```
3064     {%
```

```
3065       \@gls@link[#1]{#2}{\@glo@text}%
```

```
3066     }%
```

```
3067   }%
```

Indicate that this entry has now been used

```
3068   \ifKV@glslink@local
```

```
3069     \glslocalunset{#2}%
```

```

3070 \else
3071 \glsunset{#2}%
3072 \fi
3073 }%
3074 }

```

\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}

```

3075 \newrobustcmd*{\gls{text}}{\@ifstar\@sgls{text}\@gls{text}}

```

Define the starred form:

```

3076 \newcommand*{\@sgls{text}}[1] [] {\@gls{text}[hyper=false,#1]}

```

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

```

3077 \newcommand*{\@gls{text}}[2] [] {%
3078 \new@ifnextchar[{\@gls{text}@{#1}{#2}}{\@gls{text}@{#1}{#2} []}]

```

Read in the final optional argument:

```

3079 \def\@gls{text}@#1#2[#3] {%
3080 \glsdoifexists{#2}%
3081 {%
3082 \edef\@gls@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3083 \@gls@link[#1]{#2}{\glsentrytext{#2}#3}%
3084 }%
3085 }

```

\GLStext behaves like \gls{text} except the text is converted to uppercase.

\GLStext

```

3086 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}

```

Define the starred form:

```

3087 \newcommand*{\@sGLStext}[1] [] {\@GLStext[hyper=false,#1]}

```

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

```

3088 \newcommand*{\@GLStext}[2] [] {%
3089 \new@ifnextchar[{\@GLStext@{#1}{#2}}{\@GLStext@{#1}{#2} []}]

```

Read in the final optional argument:

```

3090 \def\@GLStext@#1#2[#3] {%
3091 \glsdoifexists{#2}%
3092 {%
3093 \edef\@gls@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3094 \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
3095 }%
3096 }

```

`\Glstext` behaves like `\glstext` except that the first letter of the text is converted to uppercase.

`\Glstext`

```
3097 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
3098 \newcommand*{\@sGlstext}[1] [] {\@Glstext[hyper=false,#1]}
```

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

```
3099 \newcommand*{\@Glstext}[2] [] {%
```

```
3100   \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3101 \def\@Glstext@#1#2[#3] {%
```

```
3102   \glsdoifexists{#2}%
```

```
3103   {%
```

```
3104     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3105     \@gls@link[#1]{#2}{\glsentrytext{#2}#3}%
```

```
3106   }%
```

```
3107 }
```

`\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`

```
3108 \newrobustcmd*{\glsfirst}{\@ifstar\@sglsfirst\@glsfirst}
```

Define the starred form:

```
3109 \newcommand*{\@sglsfirst}[1] [] {\@glsfirst[hyper=false,#1]}
```

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

```
3110 \newcommand*{\@glsfirst}[2] [] {%
```

```
3111   \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3112 \def\@glsfirst@#1#2[#3] {%
```

```
3113   \glsdoifexists{#2}%
```

```
3114   {%
```

```
3115     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3116     \@gls@link[#1]{#2}{\glsentryfirst{#2}#3}%
```

```
3117   }%
```

```
3118 }
```

`\Glsfirst` behaves like `\glsfirst` except it displays the first letter in uppercase.

`\Glsfirst`

```
3119 \newrobustcmd*{\Glsfirst}{\@ifstar\@sGlsfirst\@Glsfirst}
```

Define the starred form:

```
3120 \newcommand*{\@sGlsfirst}[1] [] {\@Glsfirst[hyper=false,#1]}
```

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

```
3121 \newcommand*{\@Glsfirst}[2] [] {%
```

```
3122   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2} []}]
```

Read in the final optional argument:

```
3123 \def\@Glsfirst@#1#2[#3] {%
```

```
3124   \glsdoifexists{#2}%
```

```
3125   {%
```

```
3126     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3127   \@gls@link[#1]{#2}{\Glsentryfirst{#2}#3}}%
```

```
3128 }
```

`\GLSfirst` behaves like `\Glsfirst` except it displays the text in uppercase.

`\GLSfirst`

```
3129 \newrobustcmd*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}
```

Define the starred form:

```
3130 \newcommand*{\@sGLSfirst}[1] [] {\@GLSfirst[hyper=false,#1]}
```

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

```
3131 \newcommand*{\@GLSfirst}[2] [] {%
```

```
3132   \new@ifnextchar[{\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2} []}]
```

Read in the final optional argument:

```
3133 \def\@GLSfirst@#1#2[#3] {%
```

```
3134   \glsdoifexists{#2}%
```

```
3135   {%
```

```
3136     \edef\@gls@type{\glsentrytype{#2}}%
```

Determine what the link text should be (this is stored in Call `\@gls@link`

```
3137   \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryfirst{#2}#3}}%
```

```
3138   }%
```

```
3139 }
```

`\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`

```
3140 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
3141 \newcommand*{\@sglsplural}[1] [] {\@glsplural[hyper=false,#1]}
```


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

```
3142 \newcommand*{\@glsplural}[2] [] {%
3143   \new@ifnextchar[{\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3144 \def\@glsplural@#1#2[#3] {%
3145   \glsdoifexists{#2}%
3146   {%
3147     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3148   \@gls@link[#1]{#2}{\glsentryplural{#2}#3}%
3149   }%
3150 }
```

\Glsplural behaves like \glsplural except that the first letter is converted to uppercase.

\Glsplural

```
3151 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}
```

Define the starred form:

```
3152 \newcommand*{\@sGlsplural}[1] [] {\@Glsplural[hyper=false,#1]}
```

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

```
3153 \newcommand*{\@Glsplural}[2] [] {%
3154   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3155 \def\@Glsplural@#1#2[#3] {%
3156   \glsdoifexists{#2}%
3157   {%
3158     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3159   \@gls@link[#1]{#2}{\Glsentryplural{#2}#3}%
3160   }%
3161 }
```

\GLSplural behaves like \glsplural except that the text is converted to uppercase.

\GLSplural

```
3162 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}
```

Define the starred form:

```
3163 \newcommand*{\@sGLSplural}[1] [] {\@GLSplural[hyper=false,#1]}
```

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

```
3164 \newcommand*{\@GLSplural}[2] [] {%
3165   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```

3166 \def\@GLSplural@#1#2[#3]{%
3167   \glsdoifexists{#2}%
3168   {%
3169     \edef\@glo@type{\glstrytype{#2}}%
3170     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glstryplural{#2}#3}}%
3171   }%
3172 }

```

Call \@gls@link

\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

```

3173 \newrobustcmd*{\glsfirstplural}{\@ifstar\@sglsfirstplural\@glsfirstplural}

```

Define the starred form:

```

3174 \newcommand*{\@sglsfirstplural}[1][\@glsfirstplural[hyper=false,#1]]{

```

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

```

3175 \newcommand*{\@glsfirstplural}[2][\@glsfirstplural@{#1}{#2}[]]{%
3176   \new@ifnextchar{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2}[]]}

```

Read in the final optional argument:

```

3177 \def\@glsfirstplural@#1#2[#3]{%
3178   \glsdoifexists{#2}%
3179   {%
3180     \edef\@glo@type{\glstrytype{#2}}%

```

Call \@gls@link

```

3181   \@gls@link[#1]{#2}{\glstryfirstplural{#2}#3}%
3182 }%
3183 }

```

\Glsfirstplural behaves like \glsfirstplural except that the first letter is converted to uppercase.

\Glsfirstplural

```

3184 \newrobustcmd*{\Glsfirstplural}{\@ifstar\@sGlsfirstplural\@Glsfirstplural}

```

Define the starred form:

```

3185 \newcommand*{\@sGlsfirstplural}[1][\@Glsfirstplural[hyper=false,#1]]{

```

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

```

3186 \newcommand*{\@Glsfirstplural}[2][\@Glsfirstplural@{#1}{#2}[]]{%
3187   \new@ifnextchar{\@Glsfirstplural@{#1}{#2}}{\@Glsfirstplural@{#1}{#2}[]]}

```

Read in the final optional argument:

```
3188 \def\@GLSfirstplural@#1#2[#3]{%
3189   \glsdoifexists{#2}%
3190   {%
3191     \edef\@glo@type{\glsentrytype{#2}}%

    Call \@gls@link
3192     \@gls@link[#1]{#2}{\Glsentryfirstplural{#2}#3}%
3193   }%
3194 }
```

`\GLSfirstplural` behaves like `\glsfirstplural` except that the link text is converted to uppercase.

`\GLSfirstplural`

```
3195 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}
```

Define the starred form:

```
3196 \newcommand*{\@sGLSfirstplural}[1][\@GLSfirstplural[hyper=false,#1]]
```

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

```
3197 \newcommand*{\@GLSfirstplural}[2][\@GLSfirstplural@{#1}{#2}[]]{%
3198   \new@ifnextchar{\@GLSfirstplural@{#1}{#2}}{\@GLSfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3199 \def\@GLSfirstplural@#1#2[#3]{%
3200   \glsdoifexists{#2}%
3201   {%
3202     \edef\@glo@type{\glsentrytype{#2}}%

    Call \@gls@link
3203     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
3204   }%
3205 }
```

`\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`

```
3206 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\@glsname}
```

Define the starred form:

```
3207 \newcommand*{\@sglsname}[1][\@glsname[hyper=false,#1]]
```

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

```
3208 \newcommand*{\@glsname}[2][\@glsname@{#1}{#2}[]]{%
3209   \new@ifnextchar{\@glsname@{#1}{#2}}{\@glsname@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3210 \def\@glsname@#1#2[#3]{%
3211   \glsdoifexists{#2}%
3212   {%
3213     \edef\@glo@type{\glsentrytype{#2}}%
3214     \@gls@link[#1]{#2}{\glsentryname{#2}#3}%
3215   }%
3216 }
```

`\Glsname` behaves like `\glsname` except that the first letter is converted to uppercase.

`\Glsname`

```
3217 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
3218 \newcommand*{\@sGlsname}[1][]{\@Glsname[hyper=false,#1]}
```

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

```
3219 \newcommand*{\@Glsname}[2][]{%
3220   \new@ifnextchar[\@Glsname@{#1}{#2}]{\@Glsname@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3221 \def\@Glsname@#1#2[#3]{%
3222   \glsdoifexists{#2}%
3223   {%
3224     \edef\@glo@type{\glsentrytype{#2}}%
3225     \@gls@link[#1]{#2}{\Glsentryname{#2}#3}%
3226   }%
3227 }
```

`\GLSname` behaves like `\glsname` except that the link text is converted to uppercase.

`\GLSname`

```
3228 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}
```

Define the starred form:

```
3229 \newcommand*{\@sGLSname}[1][]{\@GLSname[hyper=false,#1]}
```

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

```
3230 \newcommand*{\@GLSname}[2][]{%
3231   \new@ifnextchar[\@GLSname@{#1}{#2}]{\@GLSname@{#1}{#2}[]}}
```

Read in the final optional argument:

```

3232 \def\@GLSname@#1#2[#3]{%
3233   \glsdoifexists{#2}%
3234   {%
3235     \edef\@glo@type{\glsentrytype{#2}}%

    Call \@gls@link
3236     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryname{#2}#3}}%
3237   }%
3238 }
```

`\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`

```

3239 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}
```

Define the starred form:

```

3240 \newcommand*{\@sglsdesc}[1][]{\@glsdesc[hyper=false,#1]}
```

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

```

3241 \newcommand*{\@glsdesc}[2][]{%
3242   \new@ifnextchar{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```

3243 \def\@glsdesc@#1#2[#3]{%
3244   \glsdoifexists{#2}%
3245   {%
3246     \edef\@glo@type{\glsentrytype{#2}}%

    Call \@gls@link
3247     \@gls@link[#1]{#2}{\glsentrydesc{#2}#3}%
3248   }%
3249 }
```

`\Glsdesc` behaves like `\glsdesc` except that the first letter is converted to uppercase.

`\Glsdesc`

```

3250 \newrobustcmd*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}
```

Define the starred form:

```

3251 \newcommand*{\@sGlsdesc}[1][]{\@Glsdesc[hyper=false,#1]}
```

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

```

3252 \newcommand*{\@Glsdesc}[2][]{%
3253   \new@ifnextchar{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3254 \def\@GLSdesc@#1#2[#3]{%
3255   \glsdoifexists{#2}%
3256   {%
3257     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3258     \@gls@link[#1]{#2}{\glentrydesc{#2}#3}%
3259   }%
3260 }
```

\GLSdesc behaves like \glsdesc except that the link text is converted to uppercase.

\GLSdesc

```
3261 \newrobustcmd*{\@GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}
```

Define the starred form:

```
3262 \newcommand*{\@sGLSdesc}[1][]{\@GLSdesc[hyper=false,#1]}
```

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

```
3263 \newcommand*{\@GLSdesc}[2][]{%
3264   \new@ifnextchar{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3265 \def\@GLSdesc@#1#2[#3]{%
3266   \glsdoifexists{#2}%
3267   {%
3268     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3269     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentrydesc{#2}#3}}%
3270   }%
3271 }
```

\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

```
3272 \newrobustcmd*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}
```

Define the starred form:

```
3273 \newcommand*{\@sglsdescplural}[1][]{\@glsdescplural[hyper=false,#1]}
```

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

```
3274 \newcommand*{\@glsdescplural}[2][]{%
3275   \new@ifnextchar{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3276 \def\@glsdescplural@#1#2[#3]{%
3277   \glsdoifexists{#2}%
3278   {%
3279     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3280   \@gls@link[#1]{#2}{\glentrydescplural{#2}#3}%
3281   }%
3282 }
```

\Glsdescplural behaves like \glsdescplural except that the first letter is converted to uppercase.

\Glsdescplural

```
3283 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}
```

Define the starred form:

```
3284 \newcommand*{\@sGlsdescplural}[1][\@Glsdescplural[hyper=false,#1]]{
```

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

```
3285 \newcommand*{\@Glsdescplural}[2][\@Glsdescplural@{#1}{#2}[]]}
3286 \new@ifnextchar[\@Glsdescplural@{#1}{#2}]{\@Glsdescplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3287 \def\@Glsdescplural@#1#2[#3]{%
3288   \glsdoifexists{#2}%
3289   {%
3290     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3291   \@gls@link[#1]{#2}{\glentrydescplural{#2}#3}%
3292   }%
3293 }
```

\GLSdescplural behaves like \glsdescplural except that the link text is converted to uppercase.

\GLSdescplural

```
3294 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}
```

Define the starred form:

```
3295 \newcommand*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]{
```

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

```
3296 \newcommand*{\@GLSdescplural}[2][\@GLSdescplural@{#1}{#2}[]]}
3297 \new@ifnextchar[\@GLSdescplural@{#1}{#2}]{\@GLSdescplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```

3298 \def\GLSdescplural@#1#2[#3]{%
3299   \glsdoifexists{#2}%
3300   {%
3301     \edef\glo@type{\glentrytype{#2}}%
3302     Call \gls@link
3303     \gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentrydescplural{#2}#3}}%
3304   }%
3305 }
```

`\glsymbol` behaves like `\gls` except it always uses the value given by the symbol key and it doesn't mark the entry as used.

`\glsymbol`

```

3305 \newrobustcmd*{\glsymbol}{\@ifstar\sglsymbol\glsymbol}
```

Define the starred form:

```

3306 \newcommand*{\sglsymbol}[1][]{\@glsymbol[hyper=false,#1]}
```

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

```

3307 \newcommand*{\glsymbol}[2][]{%
3308   \new@ifnextchar[{\@glsymbol@{#1}{#2}}{\@glsymbol@{#1}{#2}[]}]}
```

Read in the final optional argument:

```

3309 \def\@glsymbol@#1#2[#3]{%
3310   \glsdoifexists{#2}%
3311   {%
3312     \edef\glo@type{\glentrytype{#2}}%
3313     Call \gls@link
3314     \gls@link[#1]{#2}{\glentrysymbol{#2}#3}%
3315   }%
3316 }
```

`\Glsymbol` behaves like `\glsymbol` except that the first letter is converted to uppercase.

`\Glsymbol`

```

3316 \newrobustcmd*{\Glsymbol}{\@ifstar\sglsymbol\Glsymbol}
```

Define the starred form:

```

3317 \newcommand*{\sglsymbol}[1][]{\@Glsymbol[hyper=false,#1]}
```

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

```

3318 \newcommand*{\Glsymbol}[2][]{%
3319   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2}[]}]}
```


Read in the final optional argument:

```
3320 \def\@GLssymbol@#1#2[#3]{%
3321   \glsdoifexists{#2}%
3322   {%
3323     \edef\@gls@type{\glsentrytype{#2}}%
3324     \@gls@link[#1]{#2}{\glsentrysymbol{#2}#3}%
3325   }%
3326 }
```

`\GLSsymbol` behaves like `\glsymbol` except that the link text is converted to uppercase.

`\GLSsymbol`

```
3327 \newrobustcmd*{\@GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
3328 \newcommand*{\@sGLSsymbol}[1][]{\@GLSsymbol[hyper=false,#1]}
```

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

```
3329 \newcommand*{\@GLSsymbol}[2][]{%
3330   \new@ifnextchar{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3331 \def\@GLSsymbol@#1#2[#3]{%
3332   \glsdoifexists{#2}%
3333   {%
3334     \edef\@gls@type{\glsentrytype{#2}}%
3335     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentrysymbol{#2}#3}}%
3336   }%
3337 }
```

`\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`

```
3338 \newrobustcmd*{\glsymbolplural}{\@ifstar\@sglsymbolplural\@glsymbolplural}
```

Define the starred form:

```
3339 \newcommand*{\@sglsymbolplural}[1][]{\@glsymbolplural[hyper=false,#1]}
```

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

```
3340 \newcommand*{\@glsymbolplural}[2][]{%
3341   \new@ifnextchar{\@glsymbolplural@{#1}{#2}}{\@glsymbolplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3342 \def\@glssymbolplural@#1#2[#3]{%
3343   \glsdoifexists{#2}%
3344   {%
3345     \edef\@glo@type{\glstentrytype{#2}}%
```

Call \@gls@link

```
3346     \@gls@link[#1]{#2}{\glstentrysymbolplural{#2}#3}%
3347   }%
3348 }
```

\Glsymbolplural behaves like \glssymbolplural except that the first letter is converted to uppercase.

\Glsymbolplural

```
3349 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
3350 \newcommand*{\@sGlsymbolplural}[1][\@Glsymbolplural[hyper=false,#1]]{
```

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

```
3351 \newcommand*{\@Glsymbolplural}[2][\@Glsymbolplural@{#1}{#2}[]]{%
3352   \new@ifnextchar{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2}[]]{}
```

Read in the final optional argument:

```
3353 \def\@Glsymbolplural@#1#2[#3]{%
3354   \glsdoifexists{#2}%
3355   {%
3356     \edef\@glo@type{\glstentrytype{#2}}%
```

Call \@gls@link

```
3357     \@gls@link[#1]{#2}{\glstentrysymbolplural{#2}#3}%
3358   }%
3359 }
```

\GLSsymbolplural behaves like \glssymbolplural except that the link text is converted to uppercase.

\GLSsymbolplural

```
3360 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}
```

Define the starred form:

```
3361 \newcommand*{\@sGLSsymbolplural}[1][\@GLSsymbolplural[hyper=false,#1]]{
```

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

```
3362 \newcommand*{\@GLSsymbolplural}[2][\@GLSsymbolplural@{#1}{#2}[]]{%
3363   \new@ifnextchar{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2}[]]{}
```

Read in the final optional argument:

```

3364 \def\@GLSsymbolplural@#1#2[#3]{%
3365   \glsdoifexists{#2}%
3366   {%
3367     \edef\@glo@type{\glentrytype{#2}}%
3368     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentrysymbolplural{#2}#3}}%
3369   }%
3370 }

```

`\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`

```

3371 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}

```

Define the starred form:

```

3372 \newcommand*{\@sglsuseri}[1] []{\@glsuseri[hyper=false,#1]}

```

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

```

3373 \newcommand*{\@glsuseri}[2] []{%
3374   \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} []}]

```

Read in the final optional argument:

```

3375 \def\@glsuseri@#1#2[#3]{%
3376   \glsdoifexists{#2}%
3377   {%
3378     \edef\@glo@type{\glentrytype{#2}}%

```

Call `\@gls@link`

```

3379   \@gls@link[#1]{#2}{\glentryuseri{#2}#3}%
3380 }%
3381 }

```

`\Glsuseri` behaves like `\glsuseri` except that the first letter is converted to uppercase.

`\Glsuseri`

```

3382 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}

```

Define the starred form:

```

3383 \newcommand*{\@sGlsuseri}[1] []{\@Glsuseri[hyper=false,#1]}

```

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

```

3384 \newcommand*{\@Glsuseri}[2] []{%
3385   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]

```

Read in the final optional argument:

```

3386 \def\@Glsuseri@#1#2[#3]{%
3387   \glsdoifexists{#2}%
3388   {%
3389     \edef\@glo@type{\glentrytype{#2}}%

```

Call \@gls@link

```

3390 \@gls@link[#1]{#2}{\Glsentryuseri{#2}#3}%
3391 }%
3392 }

```

\GLSuseri behaves like \glsuseri except that the link text is converted to uppercase.

\GLSuseri

```

3393 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}

```

Define the starred form:

```

3394 \newcommand*{\@sGLSuseri}[1][\@GLSuseri[hyper=false,#1]]{

```

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

```

3395 \newcommand*{\@GLSuseri}[2][\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}
3396 \new@ifnextchar[\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}

```

Read in the final optional argument:

```

3397 \def\@GLSuseri@#1#2[#3]{%
3398 \glsdoifexists{#2}%
3399 {%
3400 \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3401 \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
3402 }%
3403 }

```

\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

```

3404 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}

```

Define the starred form:

```

3405 \newcommand*{\@sglsuserii}[1][\@glsuserii[hyper=false,#1]]{

```

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

```

3406 \newcommand*{\@glsuserii}[2][\@glsuserii@{#1}{#2}]{\@glsuserii@{#1}{#2}[]}
3407 \new@ifnextchar[\@glsuserii@{#1}{#2}]{\@glsuserii@{#1}{#2}[]}

```

Read in the final optional argument:

```

3408 \def\@glsuserii@#1#2[#3]{%
3409 \glsdoifexists{#2}%
3410 {%
3411 \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3412 \@gls@link[#1]{#2}{\glsentryuserii{#2}#3}%
3413 }%
3414 }

```

`\Glsuserii` behaves like `\glsuserii` except that the first letter is converted to uppercase.

`\Glsuserii`

```
3415 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3416 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
```

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

```
3417 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3418   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3419 \def\@Glsuserii@#1#2[#3] {%
```

```
3420   \glsdoifexists{#2}%
```

```
3421   {%
```

```
3422     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3423     \@gls@link[#1]{#2}{\glsentryuserii{#2}#3}%
```

```
3424   }%
```

```
3425 }
```

`\GLSuserii` behaves like `\glsuserii` except that the link text is converted to uppercase.

`\GLSuserii`

```
3426 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}
```

Define the starred form:

```
3427 \newcommand*{\@sGLSuserii}[1] [] {\@GLSuserii[hyper=false,#1]}
```

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

```
3428 \newcommand*{\@GLSuserii}[2] [] {%
```

```
3429   \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3430 \def\@GLSuserii@#1#2[#3] {%
```

```
3431   \glsdoifexists{#2}%
```

```
3432   {%
```

```
3433     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3434     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3435   }%
```

```
3436 }
```

`\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`

```
3437 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}
```

Define the starred form:

```
3438 \newcommand*{\@sglsuseriii}[1] [] {\@glsuseriii[hyper=false,#1]}
```

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

```
3439 \newcommand*{\@glsuseriii}[2] [] {%
```

```
3440   \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3441 \def\@glsuseriii@#1#2[#3]{%
```

```
3442   \glsdoifexists{#2}%
```

```
3443   {%
```

```
3444     \edef\@glo@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3445     \@gls@link[#1]{#2}{\glsentryuseriii{#2}#3}%
```

```
3446   }%
```

```
3447 }
```

`\Glsuseriii` behaves like `\glsuseriii` except that the first letter is converted to uppercase.

`\Glsuseriii`

```
3448 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
3449 \newcommand*{\@sGlsuseriii}[1] [] {\@Glsuseriii[hyper=false,#1]}
```

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

```
3450 \newcommand*{\@Glsuseriii}[2] [] {%
```

```
3451   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3452 \def\@Glsuseriii@#1#2[#3]{%
```

```
3453   \glsdoifexists{#2}%
```

```
3454   {%
```

```
3455     \edef\@glo@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3456     \@gls@link[#1]{#2}{\Glsentryuseriii{#2}#3}%
```

```
3457   }%
```

```
3458 }
```

`\GLSuseriii` behaves like `\glsuseriii` except that the link text is converted to uppercase.

`\GLSuseriii`

```
3459 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}
```

Define the starred form:

```
3460 \newcommand*{\@sGLSuseriii}[1] [] {\@GLSuseriii[hyper=false,#1]}
```

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

```
3461 \newcommand*{\@GLSuseriii}[2] [] {%
```

```
3462   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3463 \def\@GLSuseriii@#1#2[#3] {%
```

```
3464   \glsdoifexists{#2}%
```

```
3465   {%
```

```
3466     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3467     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseriii{#2}{#3}}}%
```

```
3468   }%
```

```
3469 }
```

\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

```
3470 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
3471 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}
```

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

```
3472 \newcommand*{\@glsuseriv}[2] [] {%
```

```
3473   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3474 \def\@glsuseriv@#1#2[#3] {%
```

```
3475   \glsdoifexists{#2}%
```

```
3476   {%
```

```
3477     \edef\@gls@type{\glsentrytype{#2}}%
```

```
3478 % Call \cs{\@gls@link}
```

```
3479 % \changes{3.11a}{2013-10-15}{changed to just use \cs{\glsentryuseriv}}
```

```
3480 %   \begin{macrocode}
```

```
3481   \@gls@link[#1]{#2}{\glsentryuseriv{#2}{#3}}%
```

```
3482   }%
```

```
3483 }
```

\Glsuseriv behaves like \glsuseriv except that the first letter is converted to uppercase.

\Glsuseriv

```
3484 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
3485 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

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

```
3486 \newcommand*{\@Glsuseriv}[2] [] {%
3487   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3488 \def\@Glsuseriv@#1#2[#3] {%
3489   \glsdoifexists{#2}%
3490   {%
3491     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3492     \@gls@link[#1]{#2}{\glsentryuseriv{#2}#3}%
3493   }%
3494 }
```

\GLSuseriv behaves like \glsuseriv except that the link text is converted to uppercase.

\GLSuseriv

```
3495 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}
```

Define the starred form:

```
3496 \newcommand*{\@sGLSuseriv}[1] [] {\@GLSuseriv[hyper=false,#1]}
```

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

```
3497 \newcommand*{\@GLSuseriv}[2] [] {%
3498   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3499 \def\@GLSuseriv@#1#2[#3] {%
3500   \glsdoifexists{#2}%
3501   {%
3502     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3503     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseriv{#2}#3}}%
3504   }%
3505 }
```

\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

```
3506 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}
```

Define the starred form:

```
3507 \newcommand*{\@sglsuserv}[1] [] {\@glsuserv[hyper=false,#1]}
```

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

```
3508 \newcommand*{\@glsuserv}[2] [] {%
3509   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} [] }}
```


Read in the final optional argument:

```

3510 \def\@glsuserv@#1#2[#3]{%
3511   \glsdoifexists{#2}%
3512   {%
3513     \edef\@glo@type{\glsentrytype{#2}}%
3514     \@gls@link[#1]{#2}{\glsentryuserv{#2}#3}%
3515   }%
3516 }

```

Call \@gls@link

\Glsuserv behaves like \glsuserv except that the first letter is converted to uppercase.

\Glsuserv

```

3517 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}

```

Define the starred form:

```

3518 \newcommand*{\@sGlsuserv}[1][\@Glsuserv[hyper=false,#1]]{

```

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

```

3519 \newcommand*{\@Glsuserv}[2][\@Glsuserv@{#1}{#2}[]]{%
3520 \new@ifnextchar[\@Glsuserv@{#1}{#2}]{\@Glsuserv@{#1}{#2}[]]}

```

Read in the final optional argument:

```

3521 \def\@Glsuserv@#1#2[#3]{%
3522   \glsdoifexists{#2}%
3523   {%
3524     \edef\@glo@type{\glsentrytype{#2}}%
3525     \@gls@link[#1]{#2}{\Glsentryuserv{#2}#3}%
3526   }%
3527 }

```

\GLSuserv behaves like \glsuserv except that the link text is converted to uppercase.

\GLSuserv

```

3528 \newrobustcmd*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}

```

Define the starred form:

```

3529 \newcommand*{\@sGLSuserv}[1][\@GLSuserv[hyper=false,#1]]{

```

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

```

3530 \newcommand*{\@GLSuserv}[2][\@GLSuserv@{#1}{#2}[]]{%
3531 \new@ifnextchar[\@GLSuserv@{#1}{#2}]{\@GLSuserv@{#1}{#2}[]]}

```

Read in the final optional argument:

```

3532 \def\@GLSuserv@#1#2[#3]{%
3533   \glsdoifexists{#2}%
3534   {%
3535     \edef\@gls@type{\glsentrytype{#2}}%
3536     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuser{#2}#3}}%
3537   }%
3538 }

```

`\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`

```

3539 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}

```

Define the starred form:

```

3540 \newcommand*{\@sglsuservi}[1][]{\@glsuservi[hyper=false,#1]}

```

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

```

3541 \newcommand*{\@glsuservi}[2][]{%
3542   \new@ifnextchar[{\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2}}{}]}

```

Read in the final optional argument:

```

3543 \def\@glsuservi@#1#2[#3]{%
3544   \glsdoifexists{#2}%
3545   {%
3546     \edef\@gls@type{\glsentrytype{#2}}%
3547     \@gls@link[#1]{#2}{\glsentryuser{#2}#3}%
3548   }%
3549 }

```

`\Glsuservi` behaves like `\glsuservi` except that the first letter is converted to uppercase.

`\Glsuservi`

```

3550 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}

```

Define the starred form:

```

3551 \newcommand*{\@sGlsuservi}[1][]{\@Glsuservi[hyper=false,#1]}

```

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

```

3552 \newcommand*{\@Glsuservi}[2][]{%
3553   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2}}{}]}

```

Read in the final optional argument:

```
3554 \def\@Glsuservi@#1#2[#3]{%
3555   \glsdoifexists{#2}%
3556   {%
3557     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3558   \@gls@link[#1]{#2}{\glsentryuservi{#2}#3}%
3559 }%
3560 }
```

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

```
3561 \newrobustcmd*{\@GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}
```

Define the starred form:

```
3562 \newcommand*{\@sGLSuservi}[1][]{\@GLSuservi[hyper=false,#1]}
```

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

```
3563 \newcommand*{\@GLSuservi}[2][]{%
3564   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3565 \def\@GLSuservi@#1#2[#3]{%
3566   \glsdoifexists{#2}%
3567   {%
3568     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3569   \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuservi{#2}#3}}%
3570 }%
3571 }
```

Now deal with acronym related keys. First the short form:

\acrshort

```
3572 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\@ns@acrshort}
```

Define the starred form:

```
3573 \newcommand*{\s@acrshort}[2][]{%
3574   \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}%
3575   {\@acrshort{hyper=false,#1}{#2}[]}]%
3576 }
```

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

```
3577 \newcommand*{\ns@acrshort}[2][]{%
3578   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2}[]}]%
3579 }
```

Read in the final optional argument:

```
3580 \def\@acrshort#1#2[#3]{%
3581   \glsdoifexists{#2}%
3582   {%
3583     \edef\@glo@type{\glsentrytype{#2}}%
3584     \def\glslabel{#2}%
3585     \let\glsifplural\@secondoftwo
3586     \let\glscapscase\@firstofthree
3587     \let\glsinsert\@empty
3588     \def\glscustomtext{%
3589       \acronymfont{\glsentryshort{#2}}#3%
3590     }%
```

Call \@gls@link

```
3591   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3592   }%
3593 }
```

\Acrshort

```
3594 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\@ns@Acrshort}
```

Define the starred form:

```
3595 \newcommand*\s@Acrshort[2][{}]{%
3596   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}%
3597   {\@Acrshort{hyper=false,#1}{#2}[]}%
3598 }
```

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

```
3599 \newcommand*\@ns@Acrshort[2][{}]{%
3600   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2}[]}%
3601 }
```

Read in the final optional argument:

```
3602 \def\@Acrshort#1#2[#3]{%
3603   \glsdoifexists{#2}%
3604   {%
3605     \edef\@glo@type{\glsentrytype{#2}}%
3606     \def\glslabel{#2}%
3607     \let\glsifplural\@secondoftwo
3608     \let\glscapscase\@secondofthree
3609     \let\glsinsert\@empty
3610     \def\glscustomtext{%
3611       \acronymfont{\Glsentryshort{#2}}#3%
3612     }%
```

Call \@gls@link

```
3613   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3614   }%
3615 }
```

\ACRshort

```
3616 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
3617 \newcommand*{\s@ACRshort}[2] [] {%
3618   \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%
3619   {\@ACRshort{hyper=false,#1}{#2} []}%
3620 }
```

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

```
3621 \newcommand*{\ns@ACRshort}[2] [] {%
3622   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
3623 }
```

Read in the final optional argument:

```
3624 \def\@ACRshort#1#2[#3] {%
3625   \glsdoifexists{#2}%
3626   {%
3627     \edef\@glo@type{\glsentrytype{#2}}%
3628     \def\glslabel{#2}%
3629     \let\glsifplural\@secondoftwo
3630     \let\glscapscase\@thirdofthree
3631     \let\glsinsert\@empty
3632     \def\glscustomtext{%
3633       \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3634     }%
```

Call \@gls@link

```
3635   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3636   }%
3637 }
```

Short plural:

\acrshortpl

```
3638 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\ns@acrshortpl}
```

Define the starred form:

```
3639 \newcommand*{\s@acrshortpl}[2] [] {%
3640   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%
3641   {\@acrshortpl{hyper=false,#1}{#2} []}%
3642 }
```

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

```
3643 \newcommand*{\ns@acrshortpl}[2] [] {%
3644   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
3645 }
```

Read in the final optional argument:

```

3646 \def\@acrshortpl#1#2[#3]{%
3647   \glsdoifexists{#2}%
3648   {%
3649     \edef\@glo@type{\glsentrytype{#2}}%

3650     \def\glslabel{#2}%
3651     \let\glsifplural\@firstoftwo
3652     \let\glscapscase\@firstofthree
3653     \let\glsinsert\@empty
3654     \def\glscustomtext{%
3655       \acronymfont{\glsentryshortpl{#2}}#3%
3656     }%

```

Call \@gls@link

```

3657   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3658 }%
3659 }

```

\Acrshortpl

```

3660 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\@ns@Acrshortpl}

```

Define the starred form:

```

3661 \newcommand*\s@Acrshortpl[2][ ]{%
3662   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}%
3663   {\@Acrshortpl{hyper=false,#1}{#2}[ ]}%
3664 }

```

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

```

3665 \newcommand*\@ns@Acrshortpl[2][ ]{%
3666   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2}[ ]}%
3667 }

```

Read in the final optional argument:

```

3668 \def\@Acrshortpl#1#2[#3]{%
3669   \glsdoifexists{#2}%
3670   {%
3671     \edef\@glo@type{\glsentrytype{#2}}%

3672     \def\glslabel{#2}%
3673     \let\glsifplural\@firstoftwo
3674     \let\glscapscase\@secondofthree
3675     \let\glsinsert\@empty
3676     \def\glscustomtext{%
3677       \acronymfont{\Glsentryshortpl{#2}}#3%
3678     }%

```

Call \@gls@link

```

3679   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3680 }%
3681 }

```

\ACRshortpl

```
3682 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\@ns@ACRshortpl}
```

Define the starred form:

```
3683 \newcommand*{\s@ACRshortpl}[2] [] {%
3684   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}%
3685   {\@ACRshortpl{hyper=false,#1}{#2} []}%
3686 }
```

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

```
3687 \newcommand*{\ns@ACRshortpl}[2] [] {%
3688   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%
3689 }
```

Read in the final optional argument:

```
3690 \def\@ACRshortpl#1#2[#3] {%
3691   \glsdoifexists{#2}%
3692   {%
3693     \edef\@glo@type{\glsentrytype{#2}}%
3694     \def\glslabel{#2}%
3695     \let\glsifplural\@firstoftwo
3696     \let\glsupcase\@thirdofthree
3697     \let\glsinsert\@empty
3698     \def\glscustomtext{%
3699       \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}%
3700     }%
```

Call \@gls@link

```
3701   \@gls@link{#1}{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3702   }%
3703 }
```

\acrlong

```
3704 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\@ns@acrlong}
```

Define the starred form:

```
3705 \newcommand*{\s@acrlong}[2] [] {%
3706   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%
3707   {\@acrlong{hyper=false,#1}{#2} []}%
3708 }
```

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

```
3709 \newcommand*{\ns@acrlong}[2] [] {%
3710   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3711 }
```

Read in the final optional argument:

```

3712 \def\@acrlong#1#2[#3]{%
3713   \glsdoifexists{#2}%
3714   {%
3715     \edef\@glo@type{\glsentrytype{#2}}%

3716     \def\glslabel{#2}%
3717     \let\glsifplural\@secondoftwo
3718     \let\glscapscase\@firstofthree
3719     \let\glsinsert\@empty
3720     \def\glscustomtext{%
3721       \acronymfont{\glsentrylong{#2}}#3%
3722     }%

```

Call \@gls@link

```

3723   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3724 }%
3725 }

```

\Acrlong

```

3726 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\ns@Acrlong}

```

Define the starred form:

```

3727 \newcommand*{\s@Acrlong}[2][]{%
3728   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}%
3729   {\@Acrlong{hyper=false,#1}{#2}[]}%
3730 }

```

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

```

3731 \newcommand*{\ns@Acrlong}[2][]{%
3732   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2}[]}%
3733 }

```

Read in the final optional argument:

```

3734 \def\@Acrlong#1#2[#3]{%
3735   \glsdoifexists{#2}%
3736   {%
3737     \edef\@glo@type{\glsentrytype{#2}}%

3738     \def\glslabel{#2}%
3739     \let\glsifplural\@secondoftwo
3740     \let\glscapscase\@secondofthree
3741     \let\glsinsert\@empty
3742     \def\glscustomtext{%
3743       \acronymfont{\Glsentrylong{#2}}#3%
3744     }%

```

Call \@gls@link

```

3745   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3746 }%
3747 }

```


\ACRlong

```
3748 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\@ns@ACRlong}
```

Define the starred form:

```
3749 \newcommand*{\s@ACRlong}[2][]{%
3750   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
3751     {\@ACRlong{hyper=false,#1}{#2} []}%
3752 }
```

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

```
3753 \newcommand*{\ns@ACRlong}[2][]{%
3754   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
3755 }
```

Read in the final optional argument:

```
3756 \def\@ACRlong#1#2[#3]{%
3757   \glsdoifexists{#2}%
3758   {%
3759     \edef\@gls@type{\glsentrytype{#2}}%

3760     \def\glslabel{#2}%
3761     \let\glsifplural\@secondoftwo
3762     \let\glsupcase\@thirdofthree
3763     \let\glsinsert\@empty
3764     \def\glscustomtext{%
3765       \mfirstucMakeUppercase{\acronymfont{\glsentrylong{#2}}#3}%
3766     }%
```

Call \@gls@link

```
3767   \@gls@link[#1]{#2}{\csname gls@\@gls@type @entryfmt\endcsname}%
3768   }%
3769 }
```

Short plural:

\acrlongpl

```
3770 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\@ns@acrlongpl}
```

Define the starred form:

```
3771 \newcommand*{\s@acrlongpl}[2][]{%
3772   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
3773     {\@acrlongpl{hyper=false,#1}{#2} []}%
3774 }
```

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

```
3775 \newcommand*{\ns@acrlongpl}[2][]{%
3776   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
3777 }
```

Read in the final optional argument:

```

3778 \def\@acrlongpl#1#2[#3]{%
3779   \glsdoifexists{#2}%
3780   {%
3781     \edef\@glo@type{\glsentrytype{#2}}%
3782     \def\glslabel{#2}%
3783     \let\glsifplural\@firstoftwo
3784     \let\glscapscase\@firstofthree
3785     \let\glsinsert\@empty
3786     \def\glscustomtext{%
3787       \acronymfont{\glsentrylongpl{#2}}#3%
3788     }%

```

Call \@gls@link

```

3789   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3790 }%
3791 }

```

\Acrlongpl

```

3792 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}

```

Define the starred form:

```

3793 \newcommand*\s@Acrlongpl[2][ ]{%
3794   \new@ifnextchar[{\@Acrlongpl{hyper=false#1}{#2}}%
3795   {\@Acrlongpl{hyper=false,#1}{#2}[ ]}%
3796 }

```

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

```

3797 \newcommand*\ns@Acrlongpl[2][ ]{%
3798   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2}[ ]}%
3799 }

```

Read in the final optional argument:

```

3800 \def\@Acrlongpl#1#2[#3]{%
3801   \glsdoifexists{#2}%
3802   {%
3803     \edef\@glo@type{\glsentrytype{#2}}%
3804     \def\glslabel{#2}%
3805     \let\glsifplural\@firstoftwo
3806     \let\glscapscase\@secondofthree
3807     \let\glsinsert\@empty
3808     \def\glscustomtext{%
3809       \acronymfont{\Glsentrylongpl{#2}}#3%
3810     }%

```

Call \@gls@link

```

3811   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3812 }%
3813 }

```

\ACRlongpl

```
3814 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}
```

Define the starred form:

```
3815 \newcommand*{\s@ACRlongpl}[2] [] {%
3816   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}%
3817   {\@ACRlongpl{hyper=false,#1}{#2} []}%
3818 }
```

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

```
3819 \newcommand*{\ns@ACRlongpl}[2] [] {%
3820   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3821 }
```

Read in the final optional argument:

```
3822 \def\@ACRlongpl#1#2[#3] {%
3823   \glsdoifexists{#2}%
3824   {%
3825     \edef\@glo@type{\glsentrytype{#2}}%

3826     \def\glslabel{#2}%
3827     \let\glsifplural\@firstoftwo
3828     \let\glscapscase\@thirdofthree
3829     \let\glsinsert\@empty
3830     \def\glscustomtext{%
3831       \mfirstucMakeUppercase{\acronymfont{\glsentrylongpl{#2}}#3}%
3832     }%
```

Call \@gls@link

```
3833   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3834   }%
3835 }
```

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.

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

```
3836 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}
```

\Glsentryname

```
3837 \newrobustcmd*{\Glsentryname}[1]{%
```

```

3838 \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
3839 \expandafter\makefirstuc\expandafter{\@glo@text}%
3840 }

```

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`

```

3841 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}

```

`\Glsentrydesc`

```

3842 \newrobustcmd*{\Glsentrydesc}[1]{%
3843 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
3844 \expandafter\makefirstuc\expandafter{\@glo@text}%
3845 }

```

Plural form:

`\glsentrydescplural`

```

3846 \newcommand*{\glsentrydescplural}[1]{%
3847 \csname glo@#1@descplural\endcsname}

```

`\Glsentrydescplural`

```

3848 \newrobustcmd*{\Glsentrydescplural}[1]{%
3849 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
3850 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

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`

```

3851 \newcommand*{\glsentrytext}[1]{\csname glo@#1@text\endcsname}

```

`\Glsentrytext`

```

3852 \newrobustcmd*{\Glsentrytext}[1]{%
3853 \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
3854 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

Get the plural form:

`\glsentryplural`

```

3855 \newcommand*{\glsentryplural}[1]{\csname glo@#1@plural\endcsname}

```

`\Glsentryplural`

```

3856 \newrobustcmd*{\Glsentryplural}[1]{%
3857 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
3858 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

Get the symbol associated with this entry. The argument is the label associated with the entry. Note that unless you used `symbol=false` in the `sanitize` package option you may get unexpected results if the symbol key contained any commands.

`\glsentrysymbol`

```
3859 \newcommand*{\glsentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
3860 \newrobustcmd*{\Glsentrysymbol}[1]{%
3861 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
3862 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrysymbolplural`

```
3863 \newcommand*{\glsentrysymbolplural}[1]{%
3864 \csname glo@#1@symbolplural\endcsname}
```

`\Glsentrysymbolplural`

```
3865 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3866 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
3867 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

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`

```
3868 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

`\Glsentryfirst`

```
3869 \newrobustcmd*{\Glsentryfirst}[1]{%
3870 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
3871 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form (as specified by the `firstplural` key when the entry was defined).

`\glsentryfirstplural`

```
3872 \newcommand*{\glsentryfirstplural}[1]{%
3873 \csname glo@#1@firstpl\endcsname}
```

`\Glsentryfirstplural`

```
3874 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3875 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
3876 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Display the glossary type with which this entry is associated (as specified by the type key used when the entry was defined)

`\glsentrytype`

3877 `\newcommand*{\glsentrytype}[1]{\csname glo@#1@type\endcsname}`

Display the sort text used for this entry. Note that the sort key is sanitize, so unexpected results may occur if the sort key contained commands.

`\glsentrysort`

3878 `\newcommand*{\glsentrysort}[1]{\csname glo@#1@sort\endcsname}`

`\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.

3879 `\newcommand*{\glsentryuseri}[1]{\csname glo@#1@useri\endcsname}`

`\Glsentryuseri`

3880 `\newrobustcmd*{\Glsentryuseri}[1]{%`
3881 `\protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%`
3882 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\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.

3883 `\newcommand*{\glsentryuserii}[1]{\csname glo@#1@userii\endcsname}`

`\Glsentryuserii`

3884 `\newrobustcmd*{\Glsentryuserii}[1]{%`
3885 `\protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%`
3886 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\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.

3887 `\newcommand*{\glsentryuseriii}[1]{\csname glo@#1@useriii\endcsname}`

`\Glsentryuseriii`

3888 `\newrobustcmd*{\Glsentryuseriii}[1]{%`
3889 `\protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%`
3890 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\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.

3891 `\newcommand*{\glsentryuseriv}[1]{\csname glo@#1@useriv\endcsname}`

`\Glsentryuseriv`

3892 `\newrobustcmd*{\Glsentryuseriv}[1]{%`
3893 `\protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%`
3894 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\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.

3895 `\newcommand*{\glsentryuserv}[1]{\csname glo@#1@userv\endcsname}`

```

\Glsentryuserv
3896 \newrobustcmd*{\Glsentryuserv}[1]{%
3897 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
3898 \expandafter\makefirstuc\expandafter{\@glo@text}}

\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.
3899 \newcommand*{\glsentryuservi}[1]{\csname glo@#1@uservi\endcsname}

\Glsentryuservi
3900 \newrobustcmd*{\Glsentryuservi}[1]{%
3901 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
3902 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentryshort  Get the short key (as specified by the short the entry was defined). The argu-
                  ment is the label associated with the entry.
3903 \newcommand*{\glsentryshort}[1]{\csname glo@#1@short\endcsname}

\Glsentryshort
3904 \newrobustcmd*{\Glsentryshort}[1]{%
3905 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
3906 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentryshortpl  Get the short plural key (as specified by the shortplural the entry was defined).
                  The argument is the label associated with the entry.
3907 \newcommand*{\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}

\Glsentryshortpl
3908 \newrobustcmd*{\Glsentryshortpl}[1]{%
3909 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
3910 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentrylong  Get the long key (as specified by the long the entry was defined). The argument
                is the label associated with the entry.
3911 \newcommand*{\glsentrylong}[1]{\csname glo@#1@long\endcsname}

\Glsentrylong
3912 \newrobustcmd*{\Glsentrylong}[1]{%
3913 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
3914 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentrylongpl  Get the long plural key (as specified by the longplural the entry was defined).
                  The argument is the label associated with the entry.
3915 \newcommand*{\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}

```

`\Glsentrylongpl`

```
3916 \newrobustcmd*{\Glsentrylongpl}[1]{%
3917 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
3918 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Short cut macros to access full form:

`\glsentryfull`

```
3919 \newcommand*{\glsentryfull}[1]{%
3920 \acrfullformat{\glsentrylong{#1}}{\glsentryshort{#1}}%
3921 }
```

`\Glsentryfull`

```
3922 \newrobustcmd*{\Glsentryfull}[1]{%
3923 \acrfullformat{\Glsentrylong{#1}}{\glsentryshort{#1}}%
3924 }
```

`\glsentryfullpl`

```
3925 \newcommand*{\glsentryfullpl}[1]{%
3926 \acrfullformat{\glsentrylongpl{#1}}{\glsentryshortpl{#1}}%
3927 }
```

`\Glsentryfullpl`

```
3928 \newrobustcmd*{\Glsentryfullpl}[1]{%
3929 \acrfullformat{\Glsentrylongpl{#1}}{\glsentryshortpl{#1}}%
3930 }
```

`\glsentrynumberlist` Displays the number list as is.

```
3931 \newcommand*{\glsentrynumberlist}[1]{%
3932 \glsdoifexists{#1}%
3933 {%
3934 \csname glo@#1@numberlist\endcsname
3935 }%
3936 }
```

`\glsdisplaynumberlist` Formats the number list for the given entry label. Doesn't work with hyperref.

```
3937 \@ifpackageloaded{hyperref} {%
3938 \newcommand*{\glsdisplaynumberlist}[1]{%
3939 \GlossariesWarning
3940 {%
3941 \string\glsdisplaynumberlist\space
3942 doesn't work with hyperref.^^JUsing
3943 \string\glsentrynumberlist\space instead%
3944 }%
3945 \glsentrynumberlist{#1}%
3946 }%
3947 }%
3948 }
```



```

3949 \newcommand*{\glsdisplaynumberlist}[1]{%
3950   \glsdoifexists{#1}%
3951   {%
3952     \bgroup
3953     \def\@glo@label{#1}%
3954     \let\@org@glnumberformat\glnumberformat
3955     \def\glnumberformat##1{##1}%
3956     \protected@edef\the@numberlist{\csname glo@\@glo@label @numberlist\endcsname}%
3957     \def\@gls@numlist@sep{}%
3958     \def\@gls@numlist@nextsep{}%
3959     \def\@gls@numlist@lastsep{}%
3960     \def\@gls@thislist{}%
3961     \def\@gls@donext@def{}%
3962     \renewcommand\do[1]{%
3963       \protected@edef\@gls@thislist{%
3964         \@gls@thislist
3965         \noexpand\@gls@numlist@sep
3966         ##1%
3967       }%
3968       \let\@gls@numlist@sep\@gls@numlist@nextsep
3969       \def\@gls@numlist@nextsep{\glnumlistsep}%
3970       \@gls@donext@def
3971       \def\@gls@donext@def{%
3972         \def\@gls@numlist@lastsep{\glnumlistlastsep}%
3973       }%
3974     }%
3975     \expandafter \glnumlistparser \expandafter{\the@numberlist}%
3976     \let\@gls@numlist@sep\@gls@numlist@lastsep
3977     \@gls@thislist
3978   \egroup
3979 }%
3980 }
3981 }

```

`\glnumlistsep`

```

3982 \newcommand*{\glnumlistsep}{, }

```

`\glnumlistlastsep`

```

3983 \newcommand*{\glnumlistlastsep}{ \& }

```

`\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.

```

3984 \newcommand*{\glshyperlink}[2][\glentrytext{\@glo@label}]{%
3985 \def\@glo@label{#2}%
3986 \@glslink{\glolinkprefix#2}{#1}}

```

1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```
3987 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
3988 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
3989 \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`

```
3990 \newrobustcmd*{\glsadd}[2] [] {%
3991   \glsdoifexists{#2}%
3992   {%
3993     \def\@glsnumberformat{glsnumberformat}%
3994     \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
3995     \setkeys{glossadd}{#1}%
```

Store the entry's counter in `\theglsentrycounter`

```
3996   \@gls@saveentrycounter
3997   \@do@wrglossary{#2}%
3998 }%
3999 }
```

`\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`

```
4000 \newrobustcmd*{\glsaddall}[1] [] {%
4001   \edef\@glo@type{\@glo@types}%
4002   \setkeys{glossadd}{#1}%
4003   \forallglsentries[\@glo@type]{\@glo@entry}{%
4004     \glsadd[#1]{\@glo@entry}%
4005   }%
4006 }
```

`\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.

```

4007 \newrobustcmd*{\glsaddallunused}[1][\@glo@types]{%
4008 \forallglsentries[#1]{\@glo@entry}%
4009 {%
4010 \ifglsused{\@glo@entry}{\glsadd[format=@gobble]{\@glo@entry}}%
4011 }%
4012 }

```

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 `\glsgetgrouptitle` 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.

```

4013 \edef\glsopenbrace{\expandafter\@gobble\string\{ }

```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```

4014 \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.

```

4015 \edef\glsquote#1{\string"#1\string"}

```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for `xindy`.

```

4016 \ifglsxindy
4017 \newcommand*{\@glsfirstletter}{A}
4018 \fi

```

`stLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```

4019 \ifglsxindy

```

```

4020 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4021 \renewcommand*{\@glsfirstletter}{#1}}
4022 \else
4023 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4024 \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
4025 \fi

```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```

4026 \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.

```

4027 \ifglsxindy
4028 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4029 \renewcommand*{\@glsminrange}{#1}}
4030 \else
4031 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4032 \glsnoxywarning\GlsSetXdyMinRangeLength}
4033 \fi

```

`\writeist`

```

4034 \ifglsxindy

```

Code to use if xindy is required.

```

4035 \def\writeist{%

```

Update attributes list

```

4036 \@gls@addpredefinedattributes

```

Open the file.

```

4037 \openout\glswrite=\istfilename

```

Write header comment at the start of the file

```

4038 \write\glswrite{;; xindy style file created by the glossaries
4039 package}%
4040 \write\glswrite{;; for document '\jobname' on
4041 \the\year-\the\month-\the\day}%

```

Specify the required styles

```

4042 \write\glswrite{^^J; required styles^^J}
4043 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4044 \ifx\@xdystyle\@empty
4045 \else
4046 \protected@write\glswrite{{(require
4047 \string"\@xdystyle.xdy\string")}}%
4048 \fi
4049 }%

```

List the allowed attributes (possible values used by the format key)

```
4050 \write\glswrite{^^J%
4051 ; list of allowed attributes (number formats)^^J}%
4052 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
```

Define any additional alphabets

```
4053 \write\glswrite{^^J; user defined alphabets^^J}%
4054 \write\glswrite{\@xdyuseralphabets}%
```

Define location classes.

```
4055 \write\glswrite{^^J; location class definitions^^J}%
```

As from version 3.0, locations are now specified as $\{\langle Hprefix \rangle\}\{\langle number \rangle\}$, so need to add all possible combinations of location types.

```
4056 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
```

Case were $\langle Hprefix \rangle$ is empty:

```
4057 \protected@write\glswrite{}\{(define-location-class
4058 \string"\@gls@classI\string"^^J\space\space\space
4059 (
4060 :sep "{{"
4061 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4062 :sep "}"
4063 )
4064 ^^J\space\space\space
4065 :min-range-length \@glsminrange^^J%
4066 )
4067 }%
```

Nested iteration over all classes:

```
4068 {%
4069 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4070 \protected@write\glswrite{}\{(define-location-class
4071 \string"\@gls@classII-\@gls@classI\string"
4072 ^^J\space\space\space
4073 (
4074 :sep "{"
4075 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
4076 :sep "{{"
4077 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4078 :sep "}"
4079 )
4080 ^^J\space\space\space
4081 :min-range-length \@glsminrange^^J%
4082 )
4083 }%
4084 }%
4085 }%
4086 }%
```

User defined location classes (needs checking for new location format).

```

4087 \write\glswrite{^^J; user defined location classes}%
4088 \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.)

```

4089 \write\glswrite{^^J; define cross-reference class^^J}%
4090 \write\glswrite{(define-crossref-class \string"see\string"
4091 :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.)

```

4092 \write\glswrite{(markup-crossref-list
4093 :class \string"see\string"^^J\space\space\space
4094 :open \string"\string\glsseeformat\string"
4095 :close \string"{}\string")}%

```

List the order to sort the classes.

```

4096 \write\glswrite{^^J; define the order of the location classes}%
4097 \write\glswrite{(define-location-class-order
4098 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4099 \write\glswrite{^^J; define the glossary markup^^J}%

4100 \write\glswrite{(markup-index^^J\space\space\space
4101 :open \string"\string
4102 \glossarysection[\string\glossarytoctitle]{\string
4103 \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`)

```

4104 \@for\@this@ctr:=\@xdycounters\do{%
4105   {%
4106     \@for\@this@attr:=\@xdyattributelist\do{%
4107       \protected\write\glswrite{{}\string\providecommand*%
4108         \expandafter\string
4109         \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4110       {%
4111         \string\setentrycounter
4112         [\expandafter\@gobble\string\#1]{\@this@ctr}%
4113         \expandafter\string
4114         \csname\@this@attr\endcsname
4115         {\expandafter\@gobble\string\#2}%
4116       }%
4117     }%
4118   }%
4119 }%
4120 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```
4121 \write\glswrite{%
4122     \string\begin
4123     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4124     \space\space:close \string"\expandafter\@gobble
4125     \string~n\string~n\string
4126     \end{theglossary}\string\glossarypostamble
4127     \string~n\string" ^^J\space\space\space
4128     :tree)}}%
```

Specify what to put between letter groups

```
4129 \write\glswrite{(markup-letter-group-list
4130     :sep \string"\string\glsgroupskip\string~n\string"))}%
```

Specify what to put between entries

```
4131 \write\glswrite{(markup-indexentry
4132     :open \string"\string\relax \string\glsresetentrylist
4133     \string~n\string"))}%
```

Specify how to format entries

```
4134 \write\glswrite{(markup-locclass-list :open
4135     \string"\glsopenbrace\string\glossaryentrynumbers
4136     \glsopenbrace\string\relax\space \string"^^J\space\space\space
4137     :sep \string", \string"
4138     :close \string"\glsclosebrace\glsclosebrace\string"))}%
```

Specify how to separate location numbers

```
4139 \write\glswrite{(markup-locref-list
4140     :sep \string"\string\delimN\space\string"))}%
```

Specify how to indicate location ranges

```
4141 \write\glswrite{(markup-range
4142     :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.

```
4143 \@onelevel@sanitize\gls@suffixF
4144 \@onelevel@sanitize\gls@suffixFF
4145 \ifx\gls@suffixF\@empty
4146 \else
4147     \write\glswrite{(markup-range
4148         :close "\gls@suffixF" :length 1 :ignore-end)}}%
4149 \fi
4150 \ifx\gls@suffixFF\@empty
4151 \else
4152     \write\glswrite{(markup-range
4153         :close "\gls@suffixFF" :length 2 :ignore-end)}}%
4154 \fi
```

Specify how to format locations.

```
4155 \write\glswrite{^^J; define format to use for locations^^J}%
4156 \write\glswrite{\@xdylocref}}%
```

Specify how to separate letter groups.

```
4157 \write\glswrite{^^J; define letter group list format^^J}%
4158 \write\glswrite{(markup-letter-group-list
4159 :sep \string"\string\glsgroupskip\string~n\string")}%
```

Define letter group headings.

```
4160 \write\glswrite{^^J; letter group headings^^J}%
4161 \write\glswrite{(markup-letter-group
4162 :open-head \string"\string\glsgroupheading
4163 \glsoopenbrace\string"^^J\space\space\space
4164 :close-head \string"\glsclosebrace\string")}%
```

Define additional letter groups.

```
4165 \write\glswrite{^^J; additional letter groups^^J}%
4166 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
4167 \write\glswrite{^^J; additional sort rules^^J}
4168 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
4169 \closeout\glswrite
```

Suppress any further calls.

```
4170 \let\writeist\relax
4171 }
4172 \else
```

Code to use if makeindex is required.

```
4173 \edef\@gls@actualchar{\string?}
4174 \edef\@gls@encapchar{\string|}
4175 \edef\@gls@levelchar{\string!}
4176 \edef\@gls@quotechar{\string"}
4177 \def\writeist{\relax
4178 \openout\glswrite=\istfilename
4179 \write\glswrite{\expandafter\@gobble\string}% makeindex style file
4180 created by the glossaries package}
4181 \write\glswrite{\expandafter\@gobble\string}% for document
4182 '\jobname' on \the\year-\the\month-\the\day}
4183 \write\glswrite{actual '\@gls@actualchar'}
4184 \write\glswrite{encap '\@gls@encapchar'}
4185 \write\glswrite{level '\@gls@levelchar'}
4186 \write\glswrite{quote '\@gls@quotechar'}
4187 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4188 \write\glswrite{preamble \string"\string\glossarysection[\string
4189 \glossarytoctitle]{\string\glossarytitle}\string
4190 \glossarypreamble\string\n\string\begin{theglossary}\string
4191 \glossaryheader\string\n\string"}
4192 \write\glswrite{postamble \string"\string%\string\n\string
4193 \end{theglossary}\string\glossarypostamble\string\n
4194 \string"}
4195 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
```



```

4196     \string"}
4197 \write\glswrite{item_0 \string"\string%\string\n\string"}
4198 \write\glswrite{item_1 \string"\string%\string\n\string"}
4199 \write\glswrite{item_2 \string"\string%\string\n\string"}
4200 \write\glswrite{item_01 \string"\string%\string\n\string"}
4201 \write\glswrite{item_x1
4202     \string"\string\relax \string\glresetentrylist\string\n
4203     \string"}
4204 \write\glswrite{item_12 \string"\string%\string\n\string"}
4205 \write\glswrite{item_x2
4206     \string"\string\relax \string\glresetentrylist\string\n
4207     \string"}

4208 \write\glswrite{delim_0 \string"\string\{\string
4209     \glossaryentrynumbers\string\{\string\relax \string"}
4210 \write\glswrite{delim_1 \string"\string\{\string
4211     \glossaryentrynumbers\string\{\string\relax \string"}
4212 \write\glswrite{delim_2 \string"\string\{\string
4213     \glossaryentrynumbers\string\{\string\relax \string"}
4214 \write\glswrite{delim_t \string"\string\}\string\}\string"}
4215 \write\glswrite{delim_n \string"\string\delimN \string"}
4216 \write\glswrite{delim_r \string"\string\delimR \string"}
4217 \write\glswrite{headings_flag 1}
4218 \write\glswrite{heading_prefix
4219     \string"\string\glsgroupheading\string\{\string"}
4220 \write\glswrite{heading_suffix
4221     \string"\string\}\string\relax
4222     \string\glresetentrylist \string"}
4223 \write\glswrite{symhead_positive \string"glssymbols\string"}
4224 \write\glswrite{numhead_positive \string"glnumbers\string"}
4225 \write\glswrite{page_compositor \string"glscpositor\string"}
4226 \@glscbsdq\glscsuffixF
4227 \@glscbsdq\glscsuffixFF
4228 \ifx\glscsuffixF\empty
4229 \else
4230     \write\glswrite{suffix_2p \string"\glscsuffixF\string"}
4231 \fi
4232 \ifx\glscsuffixFF\empty
4233 \else
4234     \write\glswrite{suffix_3p \string"\glscsuffixFF\string"}
4235 \fi
4236 \closeout\glswrite
4237 \let\writeist\relax
4238 }
4239 \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`

```
4240 \newcommand{\noist}{%
```

Update attributes list

```
4241 \@gls@addpredefinedattributes
```

```
4242 \let\writeist\relax
```

```
4243 }
```

`\@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`

```
4244 \newcommand*{\@makeglossary}[1]{%
```

```
4245 \ifglossaryexists{#1}%
```

```
4246 {%
```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```
4247 \ifglssavewrites
```

```
4248 \expandafter\newtoks\csname glo@#1@filetok\endcsname
```

```
4249 \else
```

```
4250 \expandafter\newwrite\csname glo@#1@file\endcsname
```

```
4251 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
```

```
4252 \fi
```

```
4253 \@gls@renewglossary
```

```
4254 \writeist
```

```
4255 }%
```

```
4256 {%
```

```
4257 \PackageError{glossaries}%
```

```
4258 {Glossary type ‘#1’ not defined}%
```

```
4259 {New glossaries must be defined before using \string\makeglossary}%
```

```
4260 }%
```

```
4261 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
4262 \newcommand*{\@glsopenfile}[2]{%
```

```
4263 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
```

```
4264 \PackageInfo{glossaries}{Writing glossary file
```

```
4265 \jobname.\csname @glotype@#2@out\endcsname}%
```

```
4266 }
```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4267 \newcommand*{\warn@nomakeglossaries}{%
4268   \GlossariesWarningNoLine{\string\makeglossaries\space
4269   hasn't been used,^^Jthe glossaries will not be updated}%
4270 }

```

\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

```

4271 \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.
4272   \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{}%
4273   \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{}%
4274 % Write the name of the style file to the aux file
4275 % (needed by \app{makeglossaries})
4276 %   \begin{macrocode}
4277   \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
4278   \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%

```

Iterate through each glossary type and activate it.

```

4279   \@for\@glo@type:=\@glo@types\do{%
4280     \ifthenelse{\equal{\@glo@type}{}}{}{}%
4281     \@makeglossary{\@glo@type}%
4282   }%

```

New glossaries must be created before \makeglossaries so disable \newglossary.

```

4283   \renewcommand*\newglossary[4][]{%
4284   \PackageError{glossaries}{New glossaries
4285   must be created before \string\makeglossaries}{You need
4286   to move \string\makeglossaries\space after all your
4287   \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

4288   \let\@makeglossary\relax
4289   \let\makeglossary\relax
4290   \let\makeglossaries\relax

```

Disable all commands that have no effect after \makeglossaries

```

4291   \@disable@onlypremakeg

```

Allow see key:

```

4292   \let\gls@checkseeallowed\relax

```

Suppress warning about no \makeglossaries

```

4293   \let\warn@nomakeglossaries\relax

```

```

    Declare list parser for \glsdisplaynumberlist
4294 \ifglssavenumberlist
4295 \edef\@gls@dodolistparser{\noexpand\DeclareListParser
4296   {\noexpand\glsnumlistparser}{\delimN}}}%
4297 \@gls@dodolistparser
4298 \fi
4299 }

```

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`

```

4300 \let\makeglossary\makeglossaries

```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning
if neither `\printglossaries` nor `\printglossary` have been used.

```

4301 \AtEndDocument{%
4302   \warn@nomakeglossaries
4303   \warn@noprintglossary
4304 }

```

1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```

4305 \newwrite\glswrite

```

`\istfile` Deprecated.

```

4306 \def\istfile{\glswrite}

```

At the end of the document, the files should be created if `savewrites=true`.

```

4307 \AtEndDocument{%
4308   \glswritefiles
4309 }

```

`\@glswritefiles` Only write the files if `savewrites=true`

```

4310 \newcommand*\@glswritefiles{%

```

Iterate through all the glossaries

```

4311 \foralllglossaries{\@glo@type}{%

```

Check for empty glossaries (patch provided by Patrick Häcker)

```

4312   \ifcsundef{glo@\@glo@type @filetok}%
4313   {%
4314     \def\gls@tmp{%
4315     }%
4316     {%
4317       \edef\gls@tmp{\expandafter\the
4318         \csname glo@\@glo@type @filetok\endcsname}%

```

```

4319 }%
4320 \ifx\gls@tmp\@empty
4321 \ifx\@glo@type\glsdefaulttype
4322 \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
4323 entries.^^JRemember to use package option ‘nomain’ if
4324 you
4325 don’t want to^^Juse the main glossary}%
4326 \else
4327 \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
4328 entries}%
4329 \fi
4330 \else
4331 \@glsopenfile{\glswrite}{\@glo@type}%
4332 \immediate\write\glswrite{%
4333 \expandafter\the
4334 \csname glo@\@glo@type @filetok\endcsname}%
4335 \immediate\closeout\glswrite
4336 \fi
4337 }%
4338 }

```

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`

```

4339 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
4340 \@glossary[#1]%
4341 }

```

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`

```

4342 \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`

```

4343 \newcommand{\@gls@renewglossary}{%
4344 \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
4345 \let\@gls@renewglossary\@empty
4346 }

```

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

```
4347 \renewcommand*{\@wrglossary}[2]{%
4348   \ifglssavewrites
4349     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4350     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4351       \expandafter{\@gls@tmp^^J}%
4352   \else

4353     \ifcsdef{glo@#1@file}%
4354     {%
4355       \expandafter\protected@write\csname glo@#1@file\endcsname{%
4356         \gls@disablepagerefexpansion}{#2}%
4357     }%
4358     {%
4359       \GlossariesWarning{No file defined for glossary ‘#1’}%
4360     }%
4361   \fi
4362 \endgroup\@esphack
4363 }
```

\@do@wrglossary

```
4364 \newcommand*{\@do@wrglossary}[1]{%
4365   \ifglindexonlyfirst
4366     \ifglused{#1}{\@do@wrglossary{#1}}%
4367   \else
4368     \@do@wrglossary{#1}%
4369   \fi
4370 }
```

@protected@pagefmts List of page formats to be protected against expansion.

```
4371 \newcommand{\gls@protected@pagefmts}{%
4372   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4373 }
```

blepagerefexpansion

```
4374 \newcommand*{\gls@disablepagerefexpansion}{%
4375   \@for\@gls@this:=\gls@protected@pagefmts\do
4376   {%
4377     \expandafter\let\@gls@this\relax
4378   }%
4379 }
```

\gls@alphpage

```
4380 \newcommand*{\gls@alphpage}{\@alph\c@page}
```

\gls@Alphpage

```
4381 \newcommand*{\gls@Alphpage}{\@Alph\c@page}
```

```

\gls@numberpage
4382 \newcommand*{\gls@numberpage}{\number\c@page}

\gls@romanpage
4383 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

\gls@Romanpage
4384 \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.
4385 \newcommand*{\@@do@wrglossary}[1]{%
4386   \begingroup

   First a bit of hackery to prevent premature expansion of \c@page. Store original
   definitions:
4387   \let\orgthe\the
4388   \let\orgnumber\number
4389   \let\orgromannumeral\romannumeral
4390   \let\orgalph\@alph
4391   \let\orgAlph\@Alph
4392   \let\orgRoman\@Roman

   Redefine:
4393   \def\the##1{%
4394     \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4395   \def\number##1{%
4396     \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4397   \def\romannumeral##1{%
4398     \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4399   \def\@Roman##1{%
4400     \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4401   \def\@alph##1{%
4402     \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4403   \def\@Alph##1{%
4404     \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%

   Prevent expansion:
4405   \gls@disablepagerefexpansion

   Now store location in \gls@locref:
4406   \protected@xdef\gls@locref{\theHglentrycounter}%
4407   \endgroup

   Escape any special characters
4408   \@gls@checkmkidxchars\gls@locref

   Check if the hyper-location is the same as the location and set the hyper prefix.

4409   \expandafter\ifx\theHglentrycounter\theHglentrycounter

```

```

4410 \def\@glo@counterprefix{%
4411 \else
4412 \protected@edef\@glsHlocref{\theHglentrycounter}%
4413 \@gls@checkmkidxchars\@glsHlocref
4414 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4415 {\@glslocref}{\@glsHlocref}%
4416 }%
4417 \@do@gls@getcounterprefix
4418 \fi

```

Determine whether to use xindy or makeindex syntax

```

4419 \ifglsxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

4420 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
4421 \def\@glo@range{%
4422 \expandafter\if\@glo@prefix(\relax
4423 \def\@glo@range{:open-range}%
4424 \else
4425 \expandafter\if\@glo@prefix)\relax
4426 \def\@glo@range{:close-range}%
4427 \fi
4428 \fi

```

Write to the glossary file using xindy syntax.

```

4429 \glossary[\csname glo@#1@type\endcsname]{%
4430 (indexentry :tkey (\csname glo@#1@index\endcsname)
4431 :locref \string"\@glo@counterprefix}{\@glslocref}\string" %
4432 :attr \string"\@gls@counter\@glo@suffix\string"
4433 \@glo@range
4434 )
4435 }%
4436 \else

```

Convert the format information into the format required for makeindex

```

4437 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
4438 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

4439 \glossary[\csname glo@#1@type\endcsname]{%
4440 \string\glossaryentry{\csname glo@#1@index\endcsname
4441 \@gls@encapchar\@glo@numfmt}{\@glslocref}}%
4442 \fi
4443 }

```

`\ls@getcounterprefix` Get the prefix that needs to be prepended to counter in order to get the hyper counter. (For example, with the standard article class and hyperref, `\theequation` needs to be prefixed with `\section num`.) to get the equivalent

\theHequation.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```

4444 \newcommand*\@gls@getcounterprefix[2]{%
4445   \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4446   \ifx\@gls@thisloc\@gls@thisHloc
4447     \def\@glo@counterprefix{%
4448   \else
4449     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4450       \def\@glo@tmp{##2}%
4451       \ifx\@glo@tmp\@empty
4452         \def\@glo@counterprefix{%
4453   \else
4454         \def\@glo@counterprefix{##1}%
4455       \fi
4456     }%
4457     \@gls@get@counterprefix#2.#1\end@getprefix
4458   \fi
4459 }
```

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.

```

4460 \newcommand{\@do@seeglossary}[2]{%
4461   \def\@gls@xref{#2}%
4462   \@onelevel@sanitize\@gls@xref
4463   \@gls@checkmkidxchars\@gls@xref
4464   \ifglxsindy
4465     \glossary[\csname glo@#1@type\endcsname]{%
4466       (indexentry
4467         :key (\csname glo@#1@index\endcsname)
4468         :xref (\string"\@gls@xref\string")
4469         :attr \string"see\string"
4470       )
4471     }%
4472   \else
4473     \glossary[\csname glo@#1@type\endcsname]{%
4474       \string\glossaryentry{\csname glo@#1@index\endcsname
4475         \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4476   \fi
4477 }
```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4478 \def\@gls@fixbraces#1#2#3\@nil{%
4479   \ifx#2[\relax
4480     \def#1{#2#3}%
4481   \else
```

```

4482     \def#1{{#2#3}}%
4483     \fi
4484 }

\glssee   \glssee{<label>}{<cross-ref list>}
4485 \DeclareRobustCommand*\glssee[3][\seename]{%
4486     \@do@seeglossary{#2}{#1}{#3}}
4487 \newcommand*\@glssee[3][\seename]{%
4488     \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.
4489 \DeclareRobustCommand*\glsseeformat[3][\seename]{%
4490     \emph{#1} \glsseelist{#2}}

\glsseelist  \glsseelist{<list>} formats list of entry labels.
4491 \DeclareRobustCommand*\glsseelist[1]{%
    If there is only one item in the list, set the last separator to do nothing.
4492     \let\@gls@dolast\relax
    Don't display separator on the first iteration of the loop
4493     \let\@gls@donext\relax
    Iterate through the labels
4494     \@for\@gls@thislabel:=#1\do{%
    Check if on last iteration of loop
4495         \ifx\@xfor@nextelement\@nnil
4496             \@gls@dolast
4497         \else
4498             \@gls@donext
4499         \fi
    display the entry for this label
4500         \glsseeitem{\@gls@thislabel}%
    Update separators
4501         \let\@gls@dolast\glsseelastsep
4502         \let\@gls@donext\glsseesep
4503     }%
4504 }

\glsseelastsep  Separator to use between penultimate and ultimate entries in a cross-referencing
                list.
4505 \newcommand*\glsseelastsep{\space\andname\space}

\glsseesep  Separator to use between entires in a cross-referencing list.
4506 \newcommand*\glsseesep{, }

```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4507 \DeclareRobustCommand*{\glsseeitem}[1]{\glshyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glstentrytext` instead of `\glstentryname`. (To avoid problems with the name key being sanitized.)

```
4508 \newcommand*{\glsseeitemformat}[1]{\glstentrytext{#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.

```
4509 \newcommand*{\gls@save@numberlist}[1]{%
4510   \ifglssavenumberlist
4511     \toks@{#1}%
4512     \edef\@do@writeaux@info{%
4513       \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
4514     }%
4515     \@onelevel@sanitize\@do@writeaux@info
4516     \protected@write\@auxout{}\@do@writeaux@info}%
4517   \fi
4518 }
```

`\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.)

```
4519 \def\warn@noprintglossary{%
4520   \GlossariesWarningNoLine{No \string\printglossary\space
4521     or \string\printglossaries\space
4522     found.^^JThis document will not have a glossary}%
4523 }
```

`\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.

```
4524 \ifcsundef{printglossary}{}%
4525 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
4526   \GlossariesWarning{Overriding \string\printglossary}%
4527   \undef\printglossary
4528 }
```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```
4529 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
```

Set up defaults.

```
4530 \def\@glo@type{\glsdefaulttype}%
4531 \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

4532 \def\glossarytoctitle{\glossarytitle}%
4533 \let\org@glossarytitle\glossarytitle
4534 \def\@glossarystyle{}%
4535 \def\gls@dotoc@title{\gls@dotoc@title{\@glo@type}}%
```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```
4536 \let\org@glossaryentrynumbers\glossaryentrynumbers
```

Localise the effects of the optional argument

```
4537 \bgroup
```

Determine settings specified in the optional argument.

```
4538 \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)

```
4539 \ifx\glossarytitle\org@glossarytitle
4540 \else
4541 \expandafter\let\csname @glo@type\@glo@type @title\endcsname
4542 \glossarytitle
4543 \fi
```

Allow a high-level user command to indicate the current glossary

```
4544 \let\currentglossary\@glo@type
```

Enable individual number lists to be suppressed.

```
4545 \let\org@glossaryentrynumbers\glossaryentrynumbers
4546 \let\glsnonextpages\@glsnonextpages
```

Enable individual number list to be activated:

```
4547 \let\glsnextpages\@glsnextpages
```

Enable suppression of description terminators.

```
4548 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
4549 \gls@dotoc@title
```

Set the glossary style

```
4550 \@glossarystyle
```

added a way to fetch the current entry label (v3.08 updated for new `\glossentry` and `\subglossentry`):

```
4551 \let\gls@org@glossaryentryfield\glossentry
```

```

4552 \let\gls@org@glossarysubentryfield\subglossentry
4553 \renewcommand{\glossentry}[1]{%
4554   \gdef\glscurrententrylabel{##1}%
4555   \gls@org@glossaryentryfield{##1}%
4556 }%
4557 \renewcommand{\subglossentry}[2]{%
4558   \gdef\glscurrententrylabel{##2}%
4559   \gls@org@glossarysubentryfield{##1}{##2}%
4560 }%

```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```

4561 \makeatletter

```

Input the glossary file, if it exists.

```

4562 \input@{\jobname.\csname @glotype@\@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.

```

4563 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4564 {}%
4565 {\null}%

```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

4566 \ifglsxindy
4567   \ifcsundef{@xdy@\@glo@type @language}%
4568   {%
4569     \edef\@do@auxoutstuff{%
4570       \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.

```

4571         \noexpand\immediate\noexpand\write\@auxout{%
4572           \string\providecommand\string\@xdylanguage[2]{}}%
4573         \noexpand\immediate\noexpand\write\@auxout{%
4574           \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4575       }%
4576   }%
4577 }%
4578 {%
4579   \edef\@do@auxoutstuff{%
4580     \noexpand\AtEndDocument{%
4581       \noexpand\immediate\noexpand\write\@auxout{%
4582         \string\providecommand\string\@xdylanguage[2]{}}%
4583       \noexpand\immediate\noexpand\write\@auxout{%
4584         \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4585           @language\endcsname}}%
4586     }%

```

```

4587     }%
4588     }%
4589     \@do@auxoutstuff
4590     \edef\@do@auxoutstuff{%
4591         \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.

```

4592         \noexpand\immediate\noexpand\write\@auxout{%
4593             \string\providecommand\string\@gls@codepage[2]{}%
4594             \noexpand\immediate\noexpand\write\@auxout{%
4595                 \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
4596         }%
4597     }%
4598     \@do@auxoutstuff
4599     \fi
4600 \egroup

```

Reset \glossaryentrynumbers

```

4601 \global\let\glossaryentrynumbers\org@glossaryentrynumbers

```

Suppress warning about no \printglossary

```

4602 \global\let\warn@noprintglossary\relax
4603 }

```

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

```

4604 \newcommand*\printglossaries{%
4605     \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
4606 }

```

The keys that can be used in the optional argument to \printglossary are as follows: The type key sets the glossary type.

```

4607 \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.

```

4608 \define@key{printgloss}{title}{%
4609     \def\glossarytitle{#1}%
4610     \let\gls@dotoc\ttitle\relax
4611 }

```

The toctitle sets the text used for the relevant entry in the table of contents.

```
4612 \define@key{printgloss}{toctitle}{%
4613   \def\glossarytoctitle{#1}%
4614   \let\gls@dotocitle\relax
4615 }
```

The style key sets the glossary style (but only for the given glossary).

```
4616 \define@key{printgloss}{style}{%
4617   \ifcsundef{@glsstyle@#1}%
4618   {%
4619     \PackageError{glossaries}%
4620     {Glossary style ‘#1’ undefined}{}%
4621   }%
4622   {%
4623     \def\@glossarystyle{\setglossentrycompatibility
4624       \csname @glsstyle@#1\endcsname}%
4625   }%
4626 }
```

The numberedsection key determines if this glossary should be in a numbered section.

```
4627 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
4628   false,nolabel,autolabel}[nolabel]{%
4629   \ifcase\nr\relax
4630     \renewcommand*{\@glossarysecstar}{*}%
4631     \renewcommand*{\@glossaryseclabel}{}%
4632   \or
4633     \renewcommand*{\@glossarysecstar}{}%
4634     \renewcommand*{\@glossaryseclabel}{}%
4635   \or
4636     \renewcommand*{\@glossarysecstar}{}%
4637     \renewcommand*{\@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
4638   \fi}
```

The nogroupskip key determines whether or not there should be a vertical gap between glossary groups.

```
4639 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
4640   \csuse{glsnogroupskip#1}%
4641 }
```

The nonumberlist key determines if this glossary should have a number list.

```
4642 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
4643   \ifglsnonumberlist
4644     \def\glossaryentrynumbers##1{%
4645   \else
4646     \def\glossaryentrynumbers##1{##1}%
4647   \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.

```
4648 \newcommand*{\@glsnonextpages}{%
4649   \gdef\glossaryentrynumbers##1{%
4650     \glsresetentrylist
4651   }%
4652 }
```

`\@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.

```
4653 \newcommand*{\@glsnextpages}{%
4654   \gdef\glossaryentrynumbers##1{%
4655     ##1\glsresetentrylist}}
```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```
4656 \newcommand*{\glsresetentrylist}{%
4657   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}
```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```
4658 \newcommand*{\glsnonextpages}{}%
```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```
4659 \newcommand*{\glsnextpages}{}%
```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```
4660 \ifglentrycounter
4661   \ifx\@gls@counterwithin\@empty
4662     \newcounter{glossaryentry}
4663   \else
4664     \newcounter{glossaryentry}[\@gls@counterwithin]
4665   \fi
4666   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
4667 \fi
```

`glossarysubentry` If the `subentrycounter` package option has been used, define a counter to number each level 1 entry.

```
4668 \ifglsubentrycounter
4669   \ifglentrycounter
4670     \newcounter{glossarysubentry}[glossaryentry]
4671   \else
4672     \newcounter{glossarysubentry}
4673   \fi
```



```

4674 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
4675 \fi

```

`\glsresetsubentrycounter` Resets the glossarysubentry counter.

```

4676 \ifglssubentrycounter
4677 \newcommand*{\glsresetsubentrycounter}{%
4678 \setcounter{glossarysubentry}{0}%
4679 }
4680 \else
4681 \newcommand*{\glsresetsubentrycounter}{}
4682 \fi

```

`\glsresetentrycounter` Resets the glossaryentry counter.

```

4683 \ifglentrycounter
4684 \newcommand*{\glsresetentrycounter}{%
4685 \setcounter{glossaryentry}{0}%
4686 }
4687 \else
4688 \newcommand*{\glsresetentrycounter}{}
4689 \fi

```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

4690 \ifglentrycounter
4691 \newcommand*{\glsstepentry}[1]{%
4692 \refstepcounter{glossaryentry}%
4693 \label{glentry-#1}%
4694 }
4695 \else
4696 \newcommand*{\glsstepentry}[1]{}
4697 \fi

```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

4698 \ifglssubentrycounter
4699 \newcommand*{\glsstepsubentry}[1]{%
4700 \def\currentglssubentry{#1}%
4701 \refstepcounter{glossarysubentry}%
4702 \label{glentry-#1}%
4703 }
4704 \else
4705 \newcommand*{\glsstepsubentry}[1]{}
4706 \fi

```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

4707 \ifglentrycounter
4708 \newcommand*{\glsrefentry}[1]{\ref{glentry-#1}}
4709 \else

```

```

4710 \ifglssubentrycounter
4711   \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
4712 \else
4713   \newcommand*{\glsrefentry}[1]{\gls{#1}}
4714 \fi
4715 \fi

```

`glsentrycounterlabel` Defines how to display the glossaryentry counter.

```

4716 \ifglssentrycounter
4717   \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
4718 \else
4719   \newcommand*{\glsentrycounterlabel}{}
4720 \fi

```

`glssubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

4721 \ifglssubentrycounter
4722   \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry)\space}
4723 \else
4724   \newcommand*{\glssubentrycounterlabel}{}
4725 \fi

```

`\glsentryitem` Step and display glossaryentry counter, if appropriate.

```

4726 \ifglssentrycounter
4727   \newcommand*{\glsentryitem}[1]{%
4728     \glsstepentry{#1}\glsentrycounterlabel
4729   }
4730 \else
4731   \newcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
4732 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

4733 \ifglssubentrycounter
4734   \newcommand*{\glssubentryitem}[1]{%
4735     \glsstepsubentry{#1}\glssubentrycounterlabel
4736   }
4737 \else
4738   \newcommand*{\glssubentryitem}[1]{}
4739 \fi

```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

4740 \ifcsundef{theglossary}%
4741 {%
4742   \newenvironment{theglossary}{}{}%
4743 }%
4744 {%
4745   \GlossariesWarning{overriding ‘theglossary’ environment}%
4746   \renewenvironment{theglossary}{}{}%
4747 }

```

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 `\glossary` 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`

```
4748 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```
4749 \newcommand*{\glstarget}[2]{\@glstarget{\glo@linkprefix#1}{#2}}

```

As from version 3.08, glossary information is now written to the external files using `\glossentry` and `\subglossentry` instead of `\glossaryentryfield` and `\glossarysubentryfield`. The default definition provides backward compatibility for glossary styles that use the old forms.

`\compatibleglossentry`

`\glossentry{<label>}{<page-list>}`

```
4750 \providecommand*{\compatibleglossentry}[2]{%
4751   \toks@{#2}%
4752   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
4753     {\noexpand\glsnamefont
4754       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
4755     {\expandafter\expandonce\csname glo@#1@desc\endcsname}}%
4756     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}}%
4757     {\the\toks@}%
4758   }%
4759   \@do@glossentry
4760 }

```

`\glossentryname`

```
4761 \newcommand*{\glossentryname}[1]{%
4762   \glsdoifexists{#1}%
4763   {%
4764     \letcs{\glo@name}{glo@#1@name}%
4765     \expandafter\glsnamefont\expandafter{\glo@name}%
4766   }%
4767 }

```

`\Glossentryname`

```
4768 \newcommand*{\Glossentryname}[1]{%

```

```

4769 \glsdoifexists{#1}%
4770 {%
4771     \glsnamefont{\Glsentryname{#1}}%
4772 }%
4773 }

```

\glossentrydesc

```

4774 \newcommand*{\glossentrydesc}[1]{%
4775     \glsdoifexists{#1}%
4776     {%
4777         \glsentrydesc{#1}%
4778     }%
4779 }

```

\Glossentrydesc

```

4780 \newcommand*{\Glossentrydesc}[1]{%
4781     \glsdoifexists{#1}%
4782     {%
4783         \Glsentrydesc{#1}%
4784     }%
4785 }

```

\glossentrysymbol

```

4786 \newcommand*{\glossentrysymbol}[1]{%
4787     \glsdoifexists{#1}%
4788     {%
4789         \glsentrysymbol{#1}%
4790     }%
4791 }

```

\Glossentrysymbol

```

4792 \newcommand*{\Glossentrysymbol}[1]{%
4793     \glsdoifexists{#1}%
4794     {%
4795         \Glsentrysymbol{#1}%
4796     }%
4797 }

```

compatiblesubglossentry `\subglossentry{<level>}{<label>}{<page-list>}`

```

4798 \providecommand*{\compatiblesubglossentry}[3]{%
4799     \toks@{#3}%
4800     \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
4801     {#2}%
4802     {\noexpand\glsnamefont
4803         {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
4804     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%

```

```

4805     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
4806     {\the\toks@}%
4807   }%
4808   \@do@subglossentry
4809 }

```

sentrycompatibility

```

4810 \newcommand*{\setglossentrycompatibility}{%
4811   \let\glossentry\compatibleglossentry
4812   \let\subglossentry\compatiblesubglossentry
4813 }
4814 \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.

```

4815 \newcommand{\glossaryentryfield}[5]{%
4816   \GlossariesWarning
4817   {Deprecated use of \string\glossaryentryfield.^^J
4818     I recommend you change to \string\glossentry.^^J
4819     If you've just upgraded, try removing your gls auxiliary
4820     files^^J and recompile}%
4821   \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.)

```

4822 \newcommand*{\glossarysubentryfield}[6]{%
4823   \GlossariesWarning
4824   {Deprecated use of \string\glossarysubentryfield.^^J
4825     I recommend you change to \string\subglossentry.^^J
4826     If you've just upgraded, try removing your gls auxiliary
4827     files^^J and recompile}%
4828   \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`

```
4829 \newcommand*{\glsgroupskip}{}
```

Each of the 28 glossary groups described above is preceded by a group heading. This is formatted by the command `\glsgroupheading` which takes one argument which is the *label* assigned to that group (not the title). The corresponding labels are: `glsymbols`, `glsnumbers`, `A`, ..., `Z`. Glossary styles must redefine this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
4830 \newcommand*{\glsgroupheading}[1]{}
```

It is possible to “trick” `makeindex` into treating entries as though they belong to the same group, even if the terms don’t start with the same letter, by modifying the sort key. For example, all entries belonging to one group could be defined so that the sort key starts with an `a`, while entries belonging to another group could be defined so that the sort key starts with a `b`, and so on. If you want each group to have a heading, you would then need to modify the translation control sequences `\glsgetgrouptitle` and `\glsgetgrouplabel` so that the label is translated into the required title (and vice-versa).

`\glsgetgrouptitle{<label>}`

This command produces the title for the glossary group whose label is given by `<label>`. By default, the group labelled `glsymbols` produces `\glsymbolsgroupname`, the group labelled `glsnumbers` produces `\glsnumbersgroupname` and all the other groups simply produce their label. As mentioned above, the group labels are: `glsymbols`, `glsnumbers`, `A`, ..., `Z`. If you want to redefine the group titles, you will need to redefine this command. Languages other than English may produce labels that are non-expandable, so we need to check for that otherwise it will create a “missing `\endcsname` inserted” error.

`\glsgetgrouptitle`

```
4831 \newcommand*{\glsgetgrouptitle}[1]{%
4832   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
4833   \@gls@grptitle
4834 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
4835 \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.

```
4836 \dtl@ifsingle{#1}%
4837 {%
```

```

4838 \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
4839 }%
4840 {%
4841 \ifboolexpr{test{\ifstrequal{#1}{glssymbols}}
4842             or test{\ifstrequal{#1}{glsnumbers}}}%
4843 {%
4844 \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
4845 }%
4846 {%
4847 \def#2{#1}%
4848 }%
4849 }%
4850 }

```

`\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`

```

4851 \newcommand*{\glsgetgrouplabel}[1]{%
4852 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
4853 \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`

```

4854 \newcommand*{\setentrycounter}[2][ ]{%
4855 \def\@glo@counterprefix{#1}%
4856 \ifx\@glo@counterprefix\@empty
4857 \def\@glo@counterprefix{.}%
4858 \else
4859 \def\@glo@counterprefix{.#1.}%
4860 \fi
4861 \def\glsentrycounter{#2}%
4862 }

```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```

4863 \newcommand*{\setglossarystyle}[1]{%
4864 \ifcsundef{@glsstyle@#1}%
4865 {%
4866 \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4867 }%
4868 {%

```

```

4869 \csname @glsstyle@#1\endcsname
4870 }%
4871 }

```

`\glossarystyle`

```

4872 \newcommand*{\glossarystyle}[1]{%
4873   \ifcsundef{@glsstyle@#1}%
4874   {%
4875     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4876   }%
4877   {%
4878     \GlossariesWarning
4879     {Deprecated command \string\glossarystyle.^^J
4880      I recommend you switch to \string\setglossarystyle\space unless
4881      you want to maintain backward compatibility}%
4882     \setglossentrycompatibility
4883     \csname @glsstyle@#1\endcsname

4884     \ifcsdef{@glscompstyle@#1}%
4885     {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
4886     {}%
4887   }%
4888 }

```

`\newglossarystyle` New glossary styles can be defined using:

```
\newglossarystyle{<name>}{<definition>}
```

The *<definition>* argument should redefine `theglossary`, `\glossaryheader`, `\glsgroupheading`, `\glossaryentryfield` and `\glsgroupskip` (see [subsection 1.18](#) for the definitions of predefined styles). Glossary styles should not redefine `\glossarypreamble` and `\glossarypostamble`, as the user should be able to switch between styles without affecting the pre- and postambles.

```

4889 \newcommand{\newglossarystyle}[2]{%
4890   \ifcsundef{@glsstyle@#1}%
4891   {%
4892     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
4893   }%
4894   {%
4895     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
4896   }%
4897 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

4898 \newcommand{\renewglossarystyle}[2]{%
4899   \ifcsundef{@glsstyle@#1}%
4900   {%
4901     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
4902   }%

```



```

4903  {%
4904    \csdef{@glsstyle@#1}{#2}%
4905  }%
4906 }

```

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`

```

4907 \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`

```

4908 \ifcsundef{hyperlink}%
4909 {%
4910   \def\glshypernumber#1{#1}%
4911 }%
4912 {%
4913   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
4914 }

```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```

4915 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
4916   \ifx\#1\%
4917     \else
4918       \@delimR#1\delimR\delimR\%
4919     \fi
4920   \ifx\#2\%
4921     \else

```

```

4922     #2%
4923   \fi
4924   \ifx\|#3\|%
4925   \else
4926     \@glshypernumber#3\@nil
4927   \fi
4928 }

```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\glshypernumber`).

`\@delimR`

```

4929 \def\@delimR#1\delimR #2\delimR #3\|{%
4930 \ifx\|#2\|%
4931   \@delimN{#1}%
4932 \else
4933   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
4934 \fi}

```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```

4935 \def\@delimN#1{\@delimN#1\delimN \delimN\|}
4936 \def\@delimN#1\delimN #2\delimN#3\|{%
4937 \ifx\|#3\|%
4938   \@gls@numberlink{#1}%
4939 \else
4940   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
4941 \fi
4942 }

```

The following code is modified from `hyperref's \HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```

4943 \def\@gls@numberlink#1{%
4944 \begingroup
4945   \toks@={}%
4946   \@gls@removespaces#1 \@nil
4947 \endgroup}

4948 \def\@gls@removespaces#1 #2\@nil{%
4949   \toks@=\expandafter{\the\toks@#1}%
4950   \ifx\|#2\|%
4951     \edef\x{\the\toks@}%
4952     \ifx\x\empty
4953     \else

4954       \hyperlink{\glstrycounter\@gls@counterprefix\the\toks@}%
4955         {\the\toks@}%
4956     \fi
4957   \else

```

```

4958 \gls@ReturnAfterFi{%
4959 \gls@removespaces#2\@nil
4960 }%
4961 \fi
4962 }
4963 \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`

```
4964 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```
4965 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```
4966 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

`\hyperbf`

```
4967 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

`\hypermd`

```
4968 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

`\hyperit`

```
4969 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

`\hypersl`

```
4970 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}
```

`\hyperup`

```
4971 \newcommand*\hyperup[1]{\textup{\glshypernumber{#1}}}
```

`\hypersc`

```
4972 \newcommand*\hypersc[1]{\textsc{\glshypernumber{#1}}}
```

`\hyperemph`

```
4973 \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.

```

4974 \newcommand{\oldacronym}[4][\gls@label]{%
4975   \def\gls@label{#2}%
4976   \newacronym[#4]{#1}{#2}{#3}%
4977   \ifcsundef{xspace}%
4978     {%
4979       \expandafter\edef\csname#1\endcsname{%
4980         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
4981     }%
4982   }%
4983   {%
4984     \expandafter\edef\csname#1\endcsname{%
4985       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
4986         \noexpand\gls{#1}\noexpand\xspace}%
4987     }%
4988   }%
4989 }
```

`\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrv⟩}{⟨long⟩}`

This is a quick way of defining acronyms, all it does is call `\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`

```
4990 \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.

```
4991 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```
4992 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
4993 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
4994 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
4995 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
4996 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
4997 \newrobustcmd*{\acrfull}{%
```

```
4998 \ifstar\s@acrfull\ns@acrfull
```

```
4999 }
```

```
5000 \newcommand*\s@acrfull[2] [] {%
```

```
5001 \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}%
```

```
5002 {\@acrfull{hyper=false,#1}{#2} [] }%
```

```
5003 }
```

```
5004 \newcommand*\ns@acrfull[2] [] {%
```

```
5005 \new@ifnextchar[{\@acrfull{#1}{#2}}%
```

```
5006 {\@acrfull{#1}{#2} [] }%
```

```
5007 }
```

Low-level macro:

```
5008 \def\@acrfull#1#2[#3] {%
```

```
5009 \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
```

```
5010 }
```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```

5011 \newcommand{\acrlinkfullformat}[5]{%
5012   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%
5013 }

```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```

5014 \newcommand{\acrfullformat}[2]{#1\space(#2)}

```

Default format for full acronym

```

\Acrfull
5015 \newrobustcmd*{\Acrfull}{%
5016   \@ifstar\s@Acrfull\ns@Acrfull
5017 }

5018 \newcommand*\s@Acrfull[2][]{%
5019   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}{%
5020     {\@Acrfull{hyper=false,#1}{#2}[]}%
5021 }
5022 \newcommand*\ns@Acrfull[2][]{%
5023   \new@ifnextchar[{\@Acrfull{#1}{#2}}{%
5024     {\@Acrfull{#1}{#2}[]}%
5025 }

```

Low-level macro:

```

5026 \def\@Acrfull#1#2[#3]{%
5027   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
5028 }

```

```

\ACRfull
5029 \newrobustcmd*{\ACRfull}{%
5030   \@ifstar\s@ACRfull\ns@ACRfull
5031 }

5032 \newcommand*\s@ACRfull[2][]{%
5033   \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}{%
5034     {\@ACRfull{hyper=false,#1}{#2}[]}%
5035 }
5036 \newcommand*\ns@ACRfull[2][]{%
5037   \new@ifnextchar[{\@ACRfull{#1}{#2}}{%
5038     {\@ACRfull{#1}{#2}[]}%
5039 }

```

Low-level macro:

```

5040 \def\@ACRfull#1#2[#3]{%
5041   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
5042 }

```

Plural:

\acrfullpl

```
5043 \newrobustcmd*{\acrfullpl}{%
5044   \@ifstar\s@acrfullpl\ns@acrfullpl
5045 }

5046 \newcommand*\s@acrfullpl[2][]{%
5047   \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%
5048     {\@acrfullpl{hyper=false,#1}{#2}[]}%
5049 }
5050 \newcommand*\ns@acrfullpl[2][]{%
5051   \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
5052     {\@acrfullpl{#1}{#2}[]}%
5053 }
```

Low-level macro:

```
5054 \def\@acrfullpl#1#2[#3]{%
5055   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5056 }
```

\Acrfullpl

```
5057 \newrobustcmd*{\Acrfullpl}{%
5058   \@ifstar\s@Acrfullpl\ns@Acrfullpl
5059 }

5060 \newcommand*\s@Acrfullpl[2][]{%
5061   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
5062     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
5063 }
5064 \newcommand*\ns@Acrfullpl[2][]{%
5065   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
5066     {\@Acrfullpl{#1}{#2}[]}%
5067 }
```

Low-level macro:

```
5068 \def\@Acrfullpl#1#2[#3]{%
5069   \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5070 }
```

\ACRfullpl

```
5071 \newrobustcmd*{\ACRfullpl}{%
5072   \@ifstar\s@ACRfullpl\ns@ACRfullpl
5073 }

5074 \newcommand*\s@ACRfullpl[2][]{%
5075   \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
5076     {\@ACRfullpl{hyper=false,#1}{#2}[]}%
5077 }
5078 \newcommand*\ns@ACRfullpl[2][]{%
5079   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
5080     {\@ACRfullpl{#1}{#2}[]}%
5081 }
```

Low-level macro:

```
5082 \def\@ACRfullpl#1#2[#3]{%
5083   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
5084 }
```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
5085 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
5086 \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.

```
5087 \newcommand*{\acrnameformat}[2]{\acronymfont{#1}}
```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```
5088 \newtoks\glskeylisttok
```

`\glslabeltok`

```
5089 \newtoks\glslabeltok
```

`\glsshorttok`

```
5090 \newtoks\glsshorttok
```

`\glslongtok`

```
5091 \newtoks\glslongtok
```

`\newacronymhook` Provide a hook for `\newacronym`:

```
5092 \newcommand*{\newacronymhook}{}
```

`AcronymDisplayStyle` Sets the default acronym display style for given glossary.

```
5093 \newcommand*{\SetDefaultAcronymDisplayStyle}[1]{%
5094   \def\glsentryfmt[1]{\glsentryfmt}%
5095 }
```

`DefaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```
5096 \newcommand*{\DefaultNewAcronymDef}{%
5097   \edef\@do@newglossaryentry{%
5098     \noexpand\newglossaryentry{\the\glslabeltok}%
5099     {%
5100       type=\acronymtype,%
```



```

5101     name={\the\glsshorttok},%
5102     sort={\the\glsshorttok},%
5103     text={\the\glsshorttok},%
5104     first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
5105     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5106     firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
5107                  {\noexpand\expandonce\noexpand\@glo@shortpl}},%
5108     short={\the\glsshorttok},%
5109     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5110     long={\the\glslongtok},%
5111     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5112     description={\the\glslongtok},%
5113     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

Remaining options specified by the user:

```

5114     \the\glskeylisttok
5115 }%
5116 }%
5117 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5118 \let\@org@gls@assign@plural\gls@assign@plural
5119 \let\@org@gls@assign@descplural\gls@assign@descplural
5120 \def\gls@assign@firstpl##1##2{%
5121     \@@gls@expand@field{##1}{firstpl}{##2}%
5122 }%
5123 \def\gls@assign@plural##1##2{%
5124     \@@gls@expand@field{##1}{plural}{##2}%
5125 }%
5126 \def\gls@assign@descplural##1##2{%
5127     \@@gls@expand@field{##1}{descplural}{##2}%
5128 }%
5129 \do@newglossaryentry
5130 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5131 \let\gls@assign@plural\@org@gls@assign@plural
5132 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5133 }

```

DefaultAcronymStyle Set up the default acronym style:

```

5134 \newcommand*{\SetDefaultAcronymStyle}{%

```

Set the display style:

```

5135     \@for\@gls@type:=\@glsacronymlists\do{%
5136         \SetDefaultAcronymDisplayStyle{\@gls@type}%
5137     }%

```

Set up the definition of \newacronym:

```

5138     \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).

```

5139     \ifx\@glsacronymlists\@empty

```

```

5140 \def\@glo@type{\acronymtype}%
5141 \setkeys{glossentry}{##1}%
5142 \DeclareAcronymList{\@glo@type}%
5143 \SetDefaultAcronymDisplayStyle{\@glo@type}%
5144 \fi
5145 \glskeylisttok{##1}%
5146 \glslabeltok{##2}%
5147 \glsshorttok{##3}%
5148 \gslongtok{##4}%
5149 \newacronymhook
5150 \DefaultNewAcronymDef
5151 }%
5152 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5153 }

```

`\acrfootnote` Used by the footnote acronym styles.

```

5154 \newcommand*{\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

`\acrlinkfootnote`

```

5155 \newcommand*{\acrlinkfootnote}[3]{%
5156 \footnote{\glslink{#1}{#2}{#3}}%
5157 }

```

`\acrnoflinkfootnote`

```

5158 \newcommand*{\acrnoflinkfootnote}[3]{%
5159 \footnote{#3}%
5160 }

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary for the description and footnote combination.

```

5161 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
5162 \def\glsentryfmt{#1}%
5163 \ifglssused{\glslabel}%
5164 {%
5165 \acronymfont{\glsentryfmt}%
5166 }%
5167 {%
5168 \firstacronymfont{\glsentryfmt}%
5169 \ifglshassymbol{\glslabel}%
5170 {%
5171 \expandafter\protect\expandafter\acrfootnote\expandafter
5172 {\@gls@link@opts}{\@gls@link@label}%
5173 {%
5174 \glsifplural
5175 {\glsentrysymbolplural{\glslabel}}%
5176 {\glsentrysymbol{\glslabel}}%
5177 }%
5178 }%
5179 }%

```

```

5180 }%
5181 }

```

otnoteNewAcronymDef

```

5182 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
5183   \edef\@do@newglossaryentry{%
5184     \noexpand\newglossaryentry{\the\glslabeltok}%
5185     {%
5186       type=\acronymtype,%
5187       name={\noexpand\acronymfont{\the\glsshorttok}},%
5188       sort={\the\glsshorttok},%
5189       first={\the\glsshorttok},%
5190       firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5191       text={\the\glsshorttok},%
5192       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5193       short={\the\glsshorttok},%
5194       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5195       long={\the\glslongtok},%
5196       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5197       symbol={\the\glslongtok},%
5198       symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5199       \the\glskeylisttok
5200     }%
5201   }%
5202   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5203   \let\@org@gls@assign@plural\gls@assign@plural
5204   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5205   \def\gls@assign@firstpl##1##2{%
5206     \@@gls@expand@field{##1}{firstpl}{##2}%
5207   }%
5208   \def\gls@assign@plural##1##2{%
5209     \@@gls@expand@field{##1}{plural}{##2}%
5210   }%
5211   \def\gls@assign@symbolplural##1##2{%
5212     \@@gls@expand@field{##1}{symbolplural}{##2}%
5213   }%
5214   \@do@newglossaryentry
5215   \let\gls@assign@plural\@org@gls@assign@plural
5216   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5217   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5218 }

```

ootnoteAcronymStyle If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

5219 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%
5220   \renewcommand{\newacronym}[4][\]{%
5221     \ifx\@glsacronymlists\@empty

```

```

5222     \def\@glo@type{\acronymtype}%
5223     \setkeys{glossentry}{##1}%
5224     \DeclareAcronymList{\@glo@type}%
5225     \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
5226     \fi
5227     \glskeylisttok{##1}%
5228     \glslabeltok{##2}%
5229     \glsshorttok{##3}%
5230     \gslongtok{##4}%
5231     \newacronymhook
5232     \DescriptionFootnoteNewAcronymDef
5233 }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

5234 \@for\@gls@type:=\@glsacronymlists\do{%
5235     \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
5236 }%

```

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.

```

5237 \ifglsacrsmallcaps
5238     \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
5239     \renewcommand*{\acrpluralsuffix}{%
5240         \glstextup{\glspluralsuffix}}%
5241 \else
5242     \ifglsacrsmaller
5243         \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
5244     \fi
5245 \fi

```

Check for package option clash

```

5246 \ifglsacrdua
5247     \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
5248         can’t both be set}{}%
5249 \fi
5250 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

5251 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
5252     \defglsentryfmt[#1]{\glsgenentryfmt}%
5253 }

```

ionDUANewAcronymDef

```

5254 \newcommand*{\DescriptionDUANewAcronymDef}{%
5255     \edef\@do@newglossaryentry{%
5256         \noexpand\newglossaryentry{\the\glslabeltok}%

```

```

5257   {%
5258     type=\acronymtype,%
5259     name={\the\glslongtok},%
5260     sort={\the\glslongtok},%
5261     text={\the\glslongtok},%
5262     first={\the\glslongtok},%
5263     plural={\noexpand\expandonce\noexpand\@glo@longpl},%
5264     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5265     short={\the\glsshorttok},%
5266     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5267     long={\the\glslongtok},%
5268     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5269     symbol={\the\glsshorttok},%
5270     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5271     \the\glskeylisttok
5272   }%
5273 }%
5274 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5275 \let\@org@gls@assign@plural\gls@assign@plural
5276 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5277 \def\gls@assign@firstpl##1##2{%
5278   \@@gls@expand@field{##1}{firstpl}{##2}%
5279 }%
5280 \def\gls@assign@plural##1##2{%
5281   \@@gls@expand@field{##1}{plural}{##2}%
5282 }%
5283 \def\gls@assign@symbolplural##1##2{%
5284   \@@gls@expand@field{##1}{symbolplural}{##2}%
5285 }%
5286 \@do@newglossaryentry
5287 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5288 \let\gls@assign@plural\@org@gls@assign@plural
5289 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5290 }

```

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.

```

5291 \newcommand*\SetDescriptionDUAAcronymStyle{%
5292   \ifglsacrsmallcaps
5293     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
5294       can't both be set}{}%
5295   \else
5296     \ifglsacrsmaller
5297       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
5298         can't both be set}{}%
5299     \fi
5300   \fi
5301   \renewcommand{\newacronym}[4][{}]{%

```

```

5302 \ifx\@glsacronymlists\@empty
5303 \def\@glo@type{\acronymtype}%
5304 \setkeys{glossentry}{##1}%
5305 \DeclareAcronymList{\@glo@type}%
5306 \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
5307 \fi
5308 \glskeylisttok{##1}%
5309 \glslabeltok{##2}%
5310 \glsshorttok{##3}%
5311 \glslongtok{##4}%
5312 \newacronymhook
5313 \DescriptionDUANewAcronymDef
5314 }%

Set display.
5315 \@for\@gls@type:=\@glsacronymlists\do{%
5316 \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
5317 }%
5318 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

5319 \newcommand*{\SetDescriptionAcronymDisplayStyle}[1]{%
5320 \def\glsentryfmt[#1]{%
5321 \ifglsused{\glslabel}%
5322 {%
5323 \let\gls@org@insert\glsinsert
5324 \let\glsinsert\@empty
5325 \acronymfont{\glsentryfmt}\gls@org@insert
5326 }%
5327 {%
5328 \glsentryfmt
5329 \ifglsassymbol{\glslabel}%
5330 {%
5331 \glsifplural
5332 {%
5333 \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
5334 }%
5335 {%
5336 \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
5337 }%
5338 \space(\protect\firstacronymfont
5339 {\gls@symbol}
5340 {\@glo@symbol}
5341 {\@glo@symbol}
5342 {\mfirstucMakeUppercase{\@glo@symbol}}}%
5343 }%
5344 }%

```

```

5345 }%
5346 }%
5347 }

```

ptionNewAcronymDef

```

5348 \newcommand*{\DescriptionNewAcronymDef}{%
5349   \edef\@do@newglossaryentry{%
5350     \noexpand\newglossaryentry{\the\glslabeltok}%
5351     {%
5352       type=\acronymtype,%
5353       name={\noexpand
5354         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
5355       sort={\the\glsshorttok},%
5356       first={\the\glslongtok},%
5357       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5358       text={\the\glsshorttok},%
5359       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5360       short={\the\glsshorttok},%
5361       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5362       long={\the\glslongtok},%
5363       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5364       symbol={\noexpand\@glo@text},%
5365       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5366       \the\glskeylisttok}%
5367   }%
5368   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5369   \let\@org@gls@assign@plural\gls@assign@plural
5370   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5371   \def\gls@assign@firstpl##1##2{%
5372     \@@gls@expand@field{##1}{firstpl}{##2}%
5373   }%
5374   \def\gls@assign@plural##1##2{%
5375     \@@gls@expand@field{##1}{plural}{##2}%
5376   }%
5377   \def\gls@assign@symbolplural##1##2{%
5378     \@@gls@expand@field{##1}{symbolplural}{##2}%
5379   }%
5380   \@do@newglossaryentry
5381   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5382   \let\gls@assign@plural\@org@gls@assign@plural
5383   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5384 }

```

riptionAcronymStyle Option description is used, but not dua or footnote. Store long form in first key and short form in text and symbol key. The name is stored using \acrnameformat to allow the user to override the way the name is displayed in the list of acronyms.

```

5385 \newcommand*{\SetDescriptionAcronymStyle}{%
5386   \renewcommand{\newacronym}[4][\]{%

```

```

5387 \ifx\@glsacronymlists\@empty
5388 \def\@glo@type{\acronymtype}%
5389 \setkeys{glossentry}{##1}%
5390 \DeclareAcronymList{\@glo@type}%
5391 \SetDescriptionAcronymDisplayStyle{\@glo@type}%
5392 \fi
5393 \glskeylisttok{##1}%
5394 \glslabeltok{##2}%
5395 \glsshorttok{##3}%
5396 \glslongtok{##4}%
5397 \newacronymhook
5398 \DescriptionNewAcronymDef
5399 }%

```

Set display.

```

5400 \@for\@gls@type:=\@glsacronymlists\do{%
5401 \SetDescriptionAcronymDisplayStyle{\@gls@type}%
5402 }%

```

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.

```

5403 \ifglsacrsmallcaps
5404 \renewcommand{\acronymfont}[1]{\textsc{##1}}
5405 \renewcommand*{\acrpluralsuffix}{%
5406 \glstextup{\glspluralsuffix}}%
5407 \else
5408 \ifglsacrsmaller
5409 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
5410 \fi
5411 \fi
5412 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

5413 \newcommand*\SetFootnoteAcronymDisplayStyle[1]{%
5414 \defglsentryfmt[#1]{%

```

Move the inserted text outside of `\acronymfont`

```

5415 \let\gls@org@insert\glsinsert
5416 \let\glsinsert\@empty
5417 \ifglsused{\glslabel}%
5418 {%
5419 \acronymfont{\glsentryfmt}\gls@org@insert
5420 }%
5421 {%
5422 \firstacronymfont{\glsentryfmt}\gls@org@insert
5423 \ifglsashaslong{\glslabel}%
5424 {%
5425 \expandafter\protect\expandafter\acrfootnote\expandafter

```



```

5426      {\@gls@link@opts}{\@gls@link@label}%
5427      {%
5428      \glsifplural
5429      {\glsentrylongpl{\glslabel}}%
5430      {\glsentrylong{\glslabel}}%
5431      }%
5432      }%

5433      {}%
5434      }%
5435      }%
5436      }

```

otnoteNewAcronymDef

```

5437 \newcommand*{\FootnoteNewAcronymDef}{%
5438   \edef\@do@newglossaryentry{%
5439     \noexpand\newglossaryentry{\the\glslabeltok}%
5440     {%
5441       type=\acronymtype,%
5442       name={\noexpand\acronymfont{\the\glsshorttok}},%
5443       sort={\the\glsshorttok},%
5444       text={\the\glsshorttok},%
5445       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5446       first={\the\glsshorttok},%
5447       firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5448       short={\the\glsshorttok},%
5449       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5450       long={\the\glslongtok},%
5451       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5452       description={\the\glslongtok},%
5453       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5454       \the\glskeylisttok
5455     }%
5456   }%
5457   \let\@org@gls@assign@plural\gls@assign@plural
5458   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5459   \let\@org@gls@assign@descplural\gls@assign@descplural
5460   \def\gls@assign@firstpl##1##2{%
5461     \@@gls@expand@field{##1}{firstpl}{##2}%
5462   }%
5463   \def\gls@assign@plural##1##2{%
5464     \@@gls@expand@field{##1}{plural}{##2}%
5465   }%
5466   \def\gls@assign@descplural##1##2{%
5467     \@@gls@expand@field{##1}{descplural}{##2}%
5468   }%
5469   \@do@newglossaryentry
5470   \let\gls@assign@plural\@org@gls@assign@plural
5471   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5472   \let\gls@assign@descplural\@org@gls@assign@descplural

```

5473 }

`\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.

```

5474 \newcommand*\SetFootnoteAcronymStyle{%
5475   \renewcommand{\newacronym}[4][\]{%
5476     \ifx\@glsacronymlists\@empty
5477       \def\@glo@type{\acronymtype}%
5478       \setkeys{glossentry}{##1}%
5479       \DeclareAcronymList{\@glo@type}%
5480       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
5481     \fi
5482     \glskeylisttok{##1}%
5483     \glslabeltok{##2}%
5484     \glsshorttok{##3}%
5485     \gslongtok{##4}%
5486     \newacronymhook
5487     \FootnoteNewAcronymDef
5488   }%

```

Set display

```

5489   \@for\@gls@type:=\@glsacronymlists\do{%
5490     \SetFootnoteAcronymDisplayStyle{\@gls@type}%
5491   }%

```

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.

```

5492   \ifglsacrsmallcaps
5493     \renewcommand*\acronymfont[1]{\textsc{##1}}%
5494     \renewcommand*\acrpluralsuffix{%
5495       \glstextup{\glspluralsuffix}}%
5496   \else
5497     \ifglsacrsmaller
5498       \renewcommand*\acronymfont[1]{\textsmaller{##1}}%
5499     \fi
5500   \fi

```

Check for option clash

```

5501   \ifglsacrdua
5502     \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
5503       can’t both be set}{}%
5504   \fi
5505 }%

```

`\lsdoparenifnotempty` 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.

```

5506 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
5507   \protected@edef\gls@tmp{#1}%
5508   \ifdefempty\gls@tmp
5509   {%
5510   {%
5511     \ifx\gls@tmp\@gls@default@value
5512     \else
5513     \space (#2{#1})%
5514     \fi
5515   }%
5516 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

5517 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
5518   \def\glsentryfmt{#1}%

Move the inserted text outside of \acronymfont
5519   \let\gls@org@insert\glsinsert
5520   \let\glsinsert\@empty
5521   \ifglsused{\glslabel}%
5522   {%
5523     \acronymfont{\glsentryfmt}\gls@org@insert
5524   }%
5525   {%
5526     \glsentryfmt
5527     \ifglshassymbol{\glslabel}%
5528     {%
5529       \glsifplural
5530       {%
5531         \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
5532       }%
5533       {%
5534         \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
5535       }%
5536       \space
5537       (\glscapscase
5538         {\firstacronymfont{\@glo@symbol}}%
5539         {\firstacronymfont{\@glo@symbol}}%
5540         {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
5541       }%
5542     {}%
5543   }%
5544 }%
5545 }

```

\SmallNewAcronymDef

```

5546 \newcommand*{\SmallNewAcronymDef}{%
5547   \edef\@do@newglossaryentry{%

```

```

5548 \noexpand\newglossaryentry{\the\glslabeltok}%
5549 {%
5550     type=\acronymtype,%
5551     name={\noexpand\acronymfont{\the\glsshorttok}},%
5552     sort={\the\glsshorttok},%
5553     text={\the\glsshorttok},%
5554     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5555     first={\the\glslongtok},%
5556     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5557     short={\the\glsshorttok},%
5558     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5559     long={\the\glslongtok},%
5560     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5561     description={\noexpand\@glo@first},%
5562     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5563     symbol={\the\glsshorttok},%
5564     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5565     \the\glskeylisttok
5566 }%
5567 }%
5568 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5569 \let\@org@gls@assign@plural\gls@assign@plural
5570 \let\@org@gls@assign@descplural\gls@assign@descplural
5571 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5572 \def\gls@assign@firstpl##1##2{%
5573     \@@gls@expand@field{##1}{firstpl}{##2}%
5574 }%
5575 \def\gls@assign@plural##1##2{%
5576     \@@gls@expand@field{##1}{plural}{##2}%
5577 }%
5578 \def\gls@assign@descplural##1##2{%
5579     \@@gls@expand@field{##1}{descplural}{##2}%
5580 }%
5581 \def\gls@assign@symbolplural##1##2{%
5582     \@@gls@expand@field{##1}{symbolplural}{##2}%
5583 }%
5584 \do@newglossaryentry
5585 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5586 \let\gls@assign@plural\@org@gls@assign@plural
5587 \let\gls@assign@descplural\@org@gls@assign@descplural
5588 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5589 }

```

etSmallAcronymStyle Neither footnote nor description required, but smallcaps or smaller specified.
 Use the symbol key to store the short form and first to store the long form.

```

5590 \newcommand*\SetSmallAcronymStyle{%
5591   \renewcommand{\newacronym}[4][\]{%
5592     \ifx\@glsacronymlists\@empty
5593       \def\@glo@type{\acronymtype}%
5594       \setkeys{glossentry}{##1}%
5595       \DeclareAcronymList{\@glo@type}%
5596       \SetSmallAcronymDisplayStyle{\@glo@type}%
5597     \fi
5598     \glskeylisttok{##1}%
5599     \glslabeltok{##2}%
5600     \glsshorttok{##3}%
5601     \gslongtok{##4}%
5602     \newacronymhook
5603     \SmallNewAcronymDef
5604   }%

```

Change the display since first only contains long form.

```

5605   \@for\@gls@type:=\@glsacronymlists\do{%
5606     \SetSmallAcronymDisplayStyle{\@gls@type}%
5607   }%

```

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.

```

5608   \ifglsacrsmallcaps
5609     \renewcommand*\acronymfont}[1]{\textsc{##1}}
5610     \renewcommand*\acrpluralsuffix{%
5611       \glstextup{\glspluralsuffix}}%
5612   \else
5613     \renewcommand*\acronymfont}[1]{\textsmaller{##1}}
5614   \fi

```

check for option clash

```

5615   \ifglsacrdua
5616     \ifglsacrsmallcaps
5617       \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
5618         can’t both be set}{}%
5619     \else
5620       \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
5621         can’t both be set}{}%
5622     \fi
5623   \fi
5624 }%

```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```

5625 \newcommand*\SetDUADisplayStyle}[1]{%
5626   \defglsentryfmt[#1]{\glsgenentryfmt}%
5627 }

```

\DUANewAcronymDef

```

5628 \newcommand*{\DUANewAcronymDef}{%
5629   \edef\@do@newglossaryentry{%
5630     \noexpand\newglossaryentry{\the\glslabeltok}%
5631     {%
5632       type=\acronymtype,%
5633       name={\the\glsshorttok},%
5634       text={\the\glslongtok},%
5635       first={\the\glslongtok},%
5636       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
5637       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5638       short={\the\glsshorttok},%
5639       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5640       long={\the\glslongtok},%
5641       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5642       description={\the\glslongtok},%
5643       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5644       symbol={\the\glsshorttok},%
5645       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5646       \the\glskeylisttok
5647     }%
5648   }%
5649   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5650   \let\@org@gls@assign@plural\gls@assign@plural
5651   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5652   \let\@org@gls@assign@descplural\gls@assign@descplural
5653   \def\gls@assign@firstpl##1##2{%
5654     \@@gls@expand@field{##1}{firstpl}{##2}%
5655   }%
5656   \def\gls@assign@plural##1##2{%
5657     \@@gls@expand@field{##1}{plural}{##2}%
5658   }%
5659   \def\gls@assign@symbolplural##1##2{%
5660     \@@gls@expand@field{##1}{symbolplural}{##2}%
5661   }%
5662   \def\gls@assign@descplural##1##2{%
5663     \@@gls@expand@field{##1}{descplural}{##2}%
5664   }%
5665   \@do@newglossaryentry
5666   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5667   \let\gls@assign@plural\@org@gls@assign@plural
5668   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5669   \let\gls@assign@descplural\@org@gls@assign@descplural
5670 }

```

\SetDUASStyle Always expand acronyms.

```

5671 \newcommand*{\SetDUASStyle}{%
5672   \renewcommand{\newacronym}[4][[]]{%
5673     \ifx\@glsacronymlists\@empty
5674       \def\@glo@type{\acronymtype}%

```

```

5675     \setkeys{glossentry}{##1}%
5676     \DeclareAcronymList{\@glo@type}%
5677     \SetDUADisplayStyle{\@glo@type}%
5678     \fi
5679     \glskeylisttok{##1}%
5680     \glslabeltok{##2}%
5681     \glsshorttok{##3}%
5682     \glslongtok{##4}%
5683     \newacronymhook
5684     \DUANewAcronymDef
5685 }%

Set the display
5686 \@for\@gls@type:=\@glsacronymlists\do{%
5687     \SetDUADisplayStyle{\@gls@type}%
5688 }%
5689 }

```

\SetAcronymStyle

```

5690 \newcommand*\SetAcronymStyle{%
5691     \SetDefaultAcronymStyle
5692     \ifglsacrdescription
5693         \ifglsacrfootnote
5694             \SetDescriptionFootnoteAcronymStyle
5695         \else
5696             \ifglsacrdua
5697                 \SetDescriptionDUAAcronymStyle
5698             \else
5699                 \SetDescriptionAcronymStyle
5700             \fi
5701         \fi
5702     \else
5703         \ifglsacrfootnote
5704             \SetFootnoteAcronymStyle
5705         \else
5706             \ifthenelse{\boolean{glsacrsmalldcaps}\OR
5707                 \boolean{glsacrsmaller}}{%
5708                 {%
5709                     \SetSmallAcronymStyle
5710                 }%
5711             }%
5712             \ifglsacrdua
5713                 \SetDUASyle
5714             \fi
5715         }%
5716     \fi
5717 \fi
5718 }

```

Set the acronym style according to the package options

5719 \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.

tCustomDisplayStyle Sets the acronym display style.

```
5720 \newcommand*{\SetCustomDisplayStyle}[1]{%
5721   \def\glentryfmt[#1]{\glsgenentryfmt}%
5722 }
```

CustomAcronymFields

```
5723 \newcommand*{\CustomAcronymFields}{%
5724   name={\the\glsshorttok},%
5725   description={\the\glslongtok},%
5726   first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
5727   firstplural={\noexpand\acrfullformat
5728     {\noexpand\glentrylongpl{\the\glslabeltok}}}%
5729     {\noexpand\glentryshortpl{\the\glslabeltok}}},%

5730   text={\the\glsshorttok},%
5731   plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
5732 }
```

CustomNewAcronymDef

```
5733 \newcommand*{\CustomNewAcronymDef}{%
5734   \protected@edef\@do@newglossaryentry{%
5735     \noexpand\newglossaryentry{\the\glslabeltok}%
5736     {%
5737       type=\acronymtype,%
5738       short={\the\glsshorttok},%
5739       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5740       long={\the\glslongtok},%
5741       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5742       user1={\the\glsshorttok},%
5743       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
5744       user3={\the\glslongtok},%
5745       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
5746       \CustomAcronymFields,%
5747       \the\glskeylisttok
5748     }%
5749   }%
5750   \@do@newglossaryentry
5751 }
```

\SetCustomStyle

```
5752 \newcommand*{\SetCustomStyle}{%
```



```

5753 \renewcommand{\newacronym}[4][]{\%
5754 \ifx\@glsacronymlists\@empty
5755 \def\@glo@type{\acronymtype}%
5756 \setkeys{glossentry}{##1}%
5757 \DeclareAcronymList{\@glo@type}%
5758 \SetCustomDisplayStyle{\@glo@type}%
5759 \fi
5760 \glskeylisttok{##1}%
5761 \glslabeltok{##2}%
5762 \glsshorttok{##3}%
5763 \gslongtok{##4}%
5764 \newacronymhook
5765 \CustomNewAcronymDef
5766 }%

Set the display
5767 \@for\@gls@type:=\@glsacronymlists\do{%
5768 \SetCustomDisplayStyle{\@gls@type}%
5769 }%
5770 }

```

fineAcronymSynonyms

```

5771 \newcommand*{\DefineAcronymSynonyms}{\%

```

Short form

\acs

```

5772 \let\acs\acrshort

```

First letter uppercase short form

\Acs

```

5773 \let\Acs\Acrshort

```

Plural short form

\acsp

```

5774 \let\acsp\acrshortpl

```

First letter uppercase plural short form

\Acsp

```

5775 \let\Acsp\Acrshortpl

```

Long form

\acl

```

5776 \let\acl\aclong

```

Plural long form

`\aclp`
 5777 `\let\aclp\acrlongpl`
 First letter upper case long form

`\Acl`
 5778 `\let\Acl\Acrlong`
 First letter upper case plural long form

`\Aclp`
 5779 `\let\Aclp\Acrlongpl`
 Full form

`\acf`
 5780 `\let\acf\acrfull`
 Plural full form

`\acfp`
 5781 `\let\acfp\acrfullpl`
 First letter upper case full form

`\Acf`
 5782 `\let\Acf\Acrfull`
 First letter upper case plural full form

`\Acfp`
 5783 `\let\Acfp\Acrfullpl`
 Standard form

`\ac`
 5784 `\let\ac\gls`
 First upper case standard form

`\Ac`
 5785 `\let\Ac\Gls`
 Standard plural form

`\acp`
 5786 `\let\acp\glspl`
 Standard first letter upper case plural form

`\Acp`
 5787 `\let\Acp\Glspl`

5788 }

Define synonyms if required

5789 \ifglsacrshortcuts

5790 \DefineAcronymSynonyms

5791 \fi

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.

5792 \RequirePackage{glossary-hypernav}

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

5793 \@gls@loadlist

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

5794 \@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.

5795 \@gls@loadsuper

The tree-like styles. These are not loaded if the `notree` package option is used.

5796 \@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.

5797 \ifx\@glossary@default@style\relax

5798 \else

5799 \setglossarystyle{\@glossary@default@style}

5800 \fi

1.19 Debugging Commands

\showgloparent \showgloparent{\label{}}

5801 \newcommand*{\showgloparent}[1]{%

5802 \expandafter\show\csname glo@#1@parent\endcsname

5803 }

\showglolevel \showglolevel{\label{}}

```

5804 \newcommand*{\showglolevel}[1]{%
5805   \expandafter\show\csname glo@#1@level\endcsname
5806 }

```

\showglolevel \showglolevel{<label>}

```

5807 \newcommand*{\showglolevel}[1]{%
5808   \expandafter\show\csname glo@#1@level\endcsname
5809 }

```

\showgloplural \showgloplural{<label>}

```

5810 \newcommand*{\showgloplural}[1]{%
5811   \expandafter\show\csname glo@#1@plural\endcsname
5812 }

```

\showglofirst \showglofirst{<label>}

```

5813 \newcommand*{\showglofirst}[1]{%
5814   \expandafter\show\csname glo@#1@first\endcsname
5815 }

```

\showglofirstpl \showglofirstpl{<label>}

```

5816 \newcommand*{\showglofirstpl}[1]{%
5817   \expandafter\show\csname glo@#1@firstpl\endcsname
5818 }

```

\showgloftype \showgloftype{<label>}

```

5819 \newcommand*{\showgloftype}[1]{%
5820   \expandafter\show\csname glo@#1@type\endcsname
5821 }

```

\showglocounter \showglocounter{<label>}

```

5822 \newcommand*{\showglocounter}[1]{%
5823   \expandafter\show\csname glo@#1@counter\endcsname
5824 }

```

\showglouserii \showglouserii{<label>}

```
5825 \newcommand*{\showglouserii}[1]{%
5826   \expandafter\show\csname glo@#1@userii\endcsname
5827 }
```

\showglouseriii \showglouseriii{<label>}

```
5828 \newcommand*{\showglouseriii}[1]{%
5829   \expandafter\show\csname glo@#1@useriii\endcsname
5830 }
```

\showglouseriv \showglouseriv{<label>}

```
5831 \newcommand*{\showglouseriv}[1]{%
5832   \expandafter\show\csname glo@#1@useriv\endcsname
5833 }
```

\showglouserv \showglouserv{<label>}

```
5834 \newcommand*{\showglouserv}[1]{%
5835   \expandafter\show\csname glo@#1@userv\endcsname
5836 }
```

\showglouservi \showglouservi{<label>}

```
5837 \newcommand*{\showglouservi}[1]{%
5838   \expandafter\show\csname glo@#1@uservi\endcsname
5839 }
```

\showgloname \showgloname{<label>}

```
5840 \newcommand*{\showgloname}[1]{%
5841   \expandafter\show\csname glo@#1@name\endcsname
5842 }
```

\showgloname \showgloname{<label>}

```
5843 \newcommand*{\showgloname}[1]{%
5844   \expandafter\show\csname glo@#1@name\endcsname
5845 }
```

`\showglodesc` `\showglodesc{<label>}`

```
5846 \newcommand*{\showglodesc}[1]{%
5847   \expandafter\show\csname glo@#1@desc\endcsname
5848 }
```

`\showglodescplural` `\showglodescplural{<label>}`

```
5849 \newcommand*{\showglodescplural}[1]{%
5850   \expandafter\show\csname glo@#1@descplural\endcsname
5851 }
```

`\showglosort` `\showglosort{<label>}`

```
5852 \newcommand*{\showglosort}[1]{%
5853   \expandafter\show\csname glo@#1@sort\endcsname
5854 }
```

`\showglosymbol` `\showglosymbol{<label>}`

```
5855 \newcommand*{\showglosymbol}[1]{%
5856   \expandafter\show\csname glo@#1@symbol\endcsname
5857 }
```

`\showglosymbolplural` `\showglosymbolplural{<label>}`

```
5858 \newcommand*{\showglosymbolplural}[1]{%
5859   \expandafter\show\csname glo@#1@symbolplural\endcsname
5860 }
```

`\showgloshort` `\showgloshort{<label>}`

```
5861 \newcommand*{\showgloshort}[1]{%
5862   \expandafter\show\csname glo@#1@short\endcsname
5863 }
```

`\showglolong` `\showglolong{<label>}`

```
5864 \newcommand*{\showglolong}[1]{%
5865   \expandafter\show\csname glo@#1@long\endcsname
5866 }
```

`\showgloindex` `\showgloindex{<label>}`

```
5867 \newcommand*{\showgloindex}[1]{%
5868   \expandafter\show\csname glo@#1@index\endcsname
5869 }
```

`\showgloflag` `\showgloflag{<label>}`

```
5870 \newcommand*{\showgloflag}[1]{%
5871   \expandafter\show\csname ifglo@#1@flag\endcsname
5872 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
5873 \newcommand*{\showacronymlists}{%
5874   \show\@glsacronymlists
5875 }
```

`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
5876 \newcommand*{\showglossaries}{%
5877   \show\@glo@types
5878 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
5879 \newcommand*{\showglossaryin}[1]{%
5880   \expandafter\show\csname @glotype@#1@in\endcsname
5881 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
5882 \newcommand*{\showglossaryout}[1]{%
5883   \expandafter\show\csname @glotype@#1@out\endcsname
5884 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
5885 \newcommand*{\showglossarytitle}[1]{%
5886   \expandafter\show\csname @glotype@#1@title\endcsname
5887 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
5888 \newcommand*{\showglossarycounter}[1]{%
5889   \expandafter\show\csname @glotype@#1@counter\endcsname
5890 }
```

`\showglossaryentries` `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
5891 \newcommand*{\showglossaryentries}[1]{%
5892   \expandafter\show\csname glolist@#1\endcsname
5893 }
```

1.20 Compatibility with version 2.07 and below

In order to fix some bugs in v3.0, it was necessary to change the way information is written to the `glo` file, which also meant a change in the format of the Xindy style file. The compatibility option is meant for documents that use a customised Xindy style file with `\noist`. With the compatibility option, hopefully xindy will still be able to process the old document, but the bugs will remain. The issues in versions 2.07 and below:

- With xindy, the counter used by the entry was hard-coded into the Xindy style file. This meant that you couldn't use the counter to swap counters.
- With both xindy and makeindex, if used with `hyperref` and `\theH{<counter>}` was different to `\thecounter`, the link in the location number would be undefined.

```
5894 \csname ifglcompatible-2.07\endcsname
5895   \RequirePackage{glossaries-compatible-207}
5896 \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.

```
5897 \NeedsTeXFormat{LaTeX2e}
```

```
5898 \ProvidesPackage{glossaries-prefix}[2013/11/14 v4.0 (NLCT)]
```

Pass all options to glossaries:

```
5899 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
5900 \ProcessOptions
```

Load glossaries:

```
5901 \RequirePackage{glossaries}
```

Add the new keys:

```
5902 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
```

```
5903 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
```

```
5904 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
```

```
5905 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%
```

Add them to \@gls@keymap:

```
5906 \appto\@gls@keymap{,%
```

```
5907   {prefixfirst}{prefixfirst},%
```

```
5908   {prefixfirstplural}{prefixfirstplural},%
```

```
5909   {prefix}{prefix},%
```

```
5910   {prefixplural}{prefixplural}}%
```

```
5911 }
```

Set the default values:

```
5912 \appto\@newglossaryentryprehook{%
```

```
5913   \def\@glo@entryprefix{}}%
```

```
5914   \def\@glo@entryprefixplural{}}%
```

```
5915   \let\@glo@entryprefixfirst\@gls@default@value
```

```
5916   \let\@glo@entryprefixfirstplural\@gls@default@value
```

```
5917 }
```

Set the assignment code:

```
5918 \appto\@newglossaryentryposthook{%
```

```
5919   \gls@assign@field{\@glo@label}{prefix}{\@glo@entryprefix}}%
```

```
5920   \gls@assign@field{\@glo@label}{prefixplural}{\@glo@entryprefixplural}}%
```

If prefixfirst has not been supplied, make it the same as prefix.

```
5921   \expandafter\gls@assign@field\expandafter
```

```
5922     {\csname glo@\@glo@label @prefix\endcsname}{\@glo@label}{prefixfirst}}%
```

```
5923     {\@glo@entryprefixfirst}}%
```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```
5924   \expandafter\gls@assign@field\expandafter
```

```
5925     {\csname glo@\@glo@label @prefixplural\endcsname}{\@glo@label}}%
```

```

5926 {prefixfirstplural}\{@glo@entryprefixfirstplural}%
5927 }

```

Define commands to access these fields:

glsentryprefixfirst

```

5928 \newcommand*{\glsentryprefixfirst}[1]{\csuse{glo@#1@prefixfirst}}

```

ryprefixfirstplural

```

5929 \newcommand*{\glsentryprefixfirstplural}[1]{\csuse{glo@#1@prefixfirstplural}}

```

\glsentryprefix

```

5930 \newcommand*{\glsentryprefix}[1]{\csuse{glo@#1@prefix}}

```

lsentryprefixplural

```

5931 \newcommand*{\glsentryprefixplural}[1]{\csuse{glo@#1@prefixplural}}

```

Now for the initial upper case variants:

Glsentryprefixfirst

```

5932 \newrobustcmd*{\Glsentryprefixfirst}[1]{%
5933   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%
5934   \xmakefirstuc\@glo@text
5935 }

```

ryprefixfirstplural

```

5936 \newrobustcmd*{\Glsentryprefixfirstplural}[1]{%
5937   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
5938   \xmakefirstuc\@glo@text
5939 }

```

\Glsentryprefix

```

5940 \newrobustcmd*{\Glsentryprefix}[1]{%
5941   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
5942   \xmakefirstuc\@glo@text
5943 }

```

lsentryprefixplural

```

5944 \newrobustcmd*{\Glsentryprefixplural}[1]{%
5945   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
5946   \xmakefirstuc\@glo@text
5947 }

```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```

5948 \newcommand*{\ifglshasprefix}[3]{%
5949   \ifcempty{glo@#1@prefix}%
5950   {#3}%
5951   {#2}%
5952 }

```

ifglshasprefixplural

```
5953 \newcommand*{\ifglshasprefixplural}[3]{%
5954   \ifcempty{glo@#1@prefixplural}%
5955   {#3}%
5956   {#2}%
5957 }
```

ifglshasprefixfirst

```
5958 \newcommand*{\ifglshasprefixfirst}[3]{%
5959   \ifcempty{glo@#1@prefixfirst}%
5960   {#3}%
5961   {#2}%
5962 }
```

asprefixfirstplural

```
5963 \newcommand*{\ifglshasprefixfirstplural}[3]{%
5964   \ifcempty{glo@#1@prefixfirstplural}%
5965   {#3}%
5966   {#2}%
5967 }
```

Define commands that insert the prefix before commands like \gls:

\pgls

```
5968 \newrobustcmd{\pgls}{\@ifstar\@spgls\@pgls}
```

\@spgls Starred version.

```
5969 \newcommand*{\@spgls}[2] [] {\@pgls@{hyper=false,#1}{#2}}
```

\@pgls Unstarred version.

```
5970 \newcommand*{\@pgls}[2] [] {%
5971   \new@ifnextchar[%
5972   {\@pgls@{#1}{#2}}%
5973   {\@pgls@{#1}{#2} []}%
5974 }
```

\@pgls@ Read in the final optional argument:

```
5975 \def\@pgls@#1#2[#3]{%
5976   \glsdoifexists{#2}%
5977   {%
5978     \ifglsused{#2}%
5979     {%
5980       \glsentryprefix{#2}%
5981     }%
5982     {%
5983       \glsentryprefixfirst{#2}%
5984     }%
5985     \@gls@{#1}{#2}[#3]%
5986   }%
5987 }
```

Similarly for the plural version:

```
\pglsp1
5988 \newrobustcmd{\pglsp1}{\@ifstar\@spglsp1\@pglsp1}

\@spglsp1  Starred version.
5989 \newcommand*{\@spglsp1}[2][\@pglsp1@{hyper=false,#1}{#2}]

\@pglsp1  Unstarred version.
5990 \newcommand*{\@pglsp1}[2][\%
5991   \new@ifnextchar[\%
5992   {\@pglsp1@{#1}{#2}}\%
5993   {\@pglsp1@{#1}{#2}}\%
5994 ]

\@pglsp1@  Read in the final optional argument:
5995 \def\@pglsp1@#1#2[#3]{\%
5996   \glsdoifexists{#2}\%
5997   {\%
5998     \ifglsused{#2}\%
5999     {\%
6000       \glsentryprefixplural{#2}\%
6001     }\%
6002     {\%
6003       \glsentryprefixfirstplural{#2}\%
6004     }\%
6005     \@glspl@{#1}{#2}[#3]\%
6006   }\%
6007 }
```

Now for the first letter upper case versions:

```
\Pgls
6008 \newrobustcmd{\Pgls}{\@ifstar\@sPgls\@Pgls}

\@sPgls  Starred version.
6009 \newcommand*{\@sPgls}[2][\@Pgls@{hyper=false,#1}{#2}]

\@Pgls  Unstarred version.
6010 \newcommand*{\@Pgls}[2][\%
6011   \new@ifnextchar[\%
6012   {\@Pgls@{#1}{#2}}\%
6013   {\@Pgls@{#1}{#2}}\%
6014 ]

\@Pgls@  Read in the final optional argument:
6015 \def\@Pgls@#1#2[#3]{\%
6016   \glsdoifexists{#2}\%
6017   {\%
```

```

6018 \ifglused{#2}%
6019 {%
6020 \ifglshasprefix{#2}%
6021 {%
6022 \Glsentryprefix{#2}%
6023 \@gls@{#1}{#2}[#3]%
6024 }%
6025 {\@Gls@{#1}{#2}[#3]}%
6026 }%
6027 {%
6028 \ifglshasprefixfirst{#2}%
6029 {%
6030 \Glsentryprefixfirst{#2}%
6031 \@gls@{#1}{#2}[#3]%
6032 }%
6033 {\@Gls@{#1}{#2}[#3]}%
6034 }%
6035 }%
6036 }

```

Similarly for the plural version:

\Pglsp1

```
6037 \newrobustcmd{\Pglsp1}{\@ifstar\@sPglsp1\@Pglsp1}
```

\@sPglsp1 Starred version.

```
6038 \newcommand*{\@sPglsp1}[2][\@Pglsp1]{hyper=false,#1}{#2}
```

\@Pglsp1 Unstarred version.

```

6039 \newcommand*{\@Pglsp1}[2][{%
6040 \new@ifnextchar[%
6041 {\@Pglsp1@{#1}{#2}}%
6042 {\@Pglsp1@{#1}{#2}[]}%
6043 }

```

\@Pglsp1@ Read in the final optional argument:

```

6044 \def\@Pglsp1@#1#2[#3]{%
6045 \glsdoifexists{#2}%
6046 {%
6047 \ifglused{#2}%
6048 {%
6049 \ifglshasprefixplural{#2}%
6050 {%
6051 \Glsentryprefixplural{#2}%
6052 \@glspl@{#1}{#2}[#3]%
6053 }%
6054 {\@Glspl@{#1}{#2}[#3]}%
6055 }%
6056 {%

```

```

6057 \ifglshasprefixfirstplural{#2}%
6058 {%
6059 \Glsentryprefixfirstplural{#2}%
6060 \@glsp1@{#1}{#2}[#3]%
6061 }%
6062 {\@Glspl@{#1}{#2}[#3]}%
6063 }%
6064 }%
6065 }

```

Finally the all upper case versions:

\PGLS

```
6066 \newrobustcmd{\PGLS}{\@ifstar\@sPGLS\PGLS}
```

\@sPGLS Starred version.

```
6067 \newcommand*{\@sPGLS}[2][\@PGLS@{hyper=false,#1}{#2}]
```

\@PGLS Unstarred version.

```

6068 \newcommand*{\@PGLS}[2][\%
6069 \new@ifnextchar[\%
6070 {\@PGLS@{#1}{#2}}%
6071 {\@PGLS@{#1}{#2}[]}%
6072 }

```

\@PGLS@ Read in the final optional argument:

```

6073 \def\@PGLS@#1#2[#3]{\%
6074 \glsdoifexists{#2}%
6075 {%
6076 \ifglsused{#2}%
6077 {%
6078 \mfirstucMakeUppercase{\glsentryprefix{#2}}%
6079 }%
6080 {%
6081 \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%
6082 }%
6083 \@GLS@{#1}{#2}[#3]%
6084 }%
6085 }

```

Plural version:

\PGLSp1

```
6086 \newrobustcmd{\PGLSp1}{\@ifstar\@sPGLSp1\PGLSp1}
```

\@sPGLSp1 Starred version.

```
6087 \newcommand*{\@sPGLSp1}[2][\@PGLSp1@{hyper=false,#1}{#2}]
```

\@PGLSp1 Unstarred version.

```
6088 \newcommand*{\@PGLSp1}[2][\{%
6089   \new@ifnextchar[%
6090   {\@PGLSp1@{#1}{#2}}}%
6091   {\@PGLSp1@{#1}{#2}[]}%
6092 }
```

\@PGLSp1@ Read in the final optional argument:

```
6093 \def\@PGLSp1@#1#2[#3]{%
6094   \glsdoifexists{#2}%
6095   {%
6096     \ifglsused{#2}%
6097     {%
6098       \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%
6099     }%
6100     {%
6101       \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%
6102     }%
6103     \@GLSp1@{#1}{#2}[#3]%
6104   }%
6105 }
```

3 Mfirstuc Documented Code

```
6106 \NeedsTeXFormat{LaTeX2e}
6107 \ProvidesPackage{mfirstuc}[2013/11/04 v1.08 (NLCT)]
```

Requires etoolbox:

```
6108 \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`.

```
6109 \newif\if@glscs
6110 \newtoks\@glsmfirst
6111 \newtoks\@glsmrest
6112 \newrobustcmd*{\makefirstuc}[1]{%
6113   \def\gls@argi{#1}%
6114   \ifx\gls@argi\@empty
```

If the argument is empty, do nothing.

```
6115   \else
```

```

6116 \def\@gls@tmp{\ #1}%
6117 \@onelevel@sanitize\@gls@tmp
6118 \expandafter\@gls@checkcs\@gls@tmp\relax\relax
6119 \if@glscs
6120 \@gls@getbody #1{}\@nil
6121 \ifx\@gls@rest\@empty
6122 \glsmakefirstuc{#1}%
6123 \else
6124 \expandafter\@gls@split\@gls@rest\@nil
6125 \ifx\@gls@first\@empty
6126 \glsmakefirstuc{#1}%
6127 \else
6128 \expandafter\@glsmfirst\expandafter{\@gls@first}%
6129 \expandafter\@glsmrest\expandafter{\@gls@rest}%
6130 \edef\@gls@domfirstuc{\noexpand\@gls@body
6131 {\noexpand\glsmakefirstuc\the\@glsmfirst}%
6132 \the\@glsmrest}%
6133 \@gls@domfirstuc
6134 \fi
6135 \fi
6136 \else
6137 \glsmakefirstuc{#1}%
6138 \fi
6139 \fi
6140 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

6141 \def\@gls@split#1#2\@nil{%
6142 \def\@gls@first{#1}\def\@gls@rest{#2}%
6143 }

6144 \def\@gls@checkcs#1 #2#3\relax{%
6145 \def\@gls@argi{#1}\def\@gls@argii{#2}%
6146 \ifx\@gls@argi\@gls@argii
6147 \@glscstrue
6148 \else
6149 \@glscsfalse
6150 \fi
6151 }

```

\@gls@makefirstuc Make first thing upper case:

```
6152 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}
```

\mfirstucMakeUppercase Allow user to replace \MakeUppercase with another case changing command.

```
6153 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}
```

\glsmakefirstuc Provide a user command to make it easier to customise.

```
6154 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}
```


Get the first grouped argument and stores in \@gls@body.

```
6155 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}
```

Scoup up everything to \@nil and store in \@gls@rest:

```
6156 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}
```

`\xmakefirstuc` Expand argument once before applying `\makefirstuc` (added v1.01).

```
6157 \newcommand*\xmakefirstuc}[1]{%
```

```
6158 \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).

```
6159 \newrobustcmd*\capitalisewords}[1]{%
```

```
6160 \def\gls@add@space{ }%
```

```
6161 \mfu@capitalisewords#1 \@nil\mfu@endcap
```

```
6162 }
```

```
6163 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
```

```
6164 \def\mfu@cap@first{#1}%
```

```
6165 \def\mfu@cap@second{#2}%
```

```
6166 \gls@add@space
```

```
6167 \makefirstuc{#1}%
```

```
6168 \def\gls@add@space{ }%
```

```
6169 \ifx\mfu@cap@second\@nnil
```

```
6170 \let\next\mfu@cap\mfu@noop
```

```
6171 \else
```

```
6172 \let\next\mfu@cap\mfu@capitalisewords
```

```
6173 \fi
```

```
6174 \next\mfu@cap#2\mfu@endcap
```

```
6175 }
```

```
6176 \def\mfu@noop#1\mfu@endcap{ }
```

`\xcapitalisewords` Short-cut command:

```
6177 \newcommand*\xcapitalisewords}[1]{%
```

```
6178 \expandafter\capitalisewords\expandafter{#1}%
```

```
6179 }
```

4 Glossary Styles

4.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
6180 \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`

```
6181 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
6182   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
6183   \@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`

```
6184 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
6185   \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
6186   \@gls@target{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
6187   \expandafter\let
6188     \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
  Iterate through list and terminate loop if this group is found.
6189   \@for\@gls@elem:=\@gls@list\do{%
6190     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}%
  Check if list terminated prematurely.
6191   \if@endfor
6192   \else
    This group was not included in the list, so issue a warning.
6193     \GlossariesWarningNoLine{Navigation panel
6194       for glossary type ‘#1’^^Jmissing group ‘#2’}%
6195     \gdef\gls@hypergroup@rerun{%
6196       \GlossariesWarningNoLine{Navigation panel
6197         has changed. Rerun LaTeX}}%
6198   \fi
6199 }
```

`\gls@hypergroup@rerun` Give a warning at the end if re-run required

```
6200 \let\gls@hypergroup@rerun\relax
6201 \AtEndDocument{\gls@hypergroup@rerun}
```

`\@gls@hypergroup` This adds to (or creates) the command `\@gls@hypergroup@#1` (*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.

```

6202 \newcommand*{\@gls@hypergroup}[2]{%
6203 \ifundefined{\@gls@hypergroup@#1}{%
6204   \expandafter\xdef\csname \@gls@hypergroup@#1\endcsname{#2}%
6205 }{%
6206   \expandafter\let\expandafter\@gls@tmp
6207     \csname \@gls@hypergroup@#1\endcsname
6208   \expandafter\xdef\csname \@gls@hypergroup@#1\endcsname{%
6209     \@gls@tmp,#2}%
6210 }%
6211 }

```

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`

```

6212 \newcommand*{\glsnavigation}{%
6213 \def\@gls@between{}%
6214 \ifundefined{\@gls@hypergroup@{\@gls@type}}{%
6215   \def\@gls@list{}%
6216 }{%
6217   \expandafter\let\expandafter\@gls@list
6218     \csname \@gls@hypergroup@{\@gls@type}\endcsname
6219 }%
6220 \@for\@gls@tmp:=\@gls@list\do{%
6221   \@gls@between
6222     \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%
6223     \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
6224     \let\@gls@between\glshypernavsep%
6225 }%
6226 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

6227 \newcommand*{\glshypernavsep}{\space\textbar\space}

```

The `\glsymbolnav` 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.

`\glsymbolnav`

```

6228 \newcommand*{\glssymbolnav}{%
6229 \glsnavigationhyperlink{glssymbols}{\glsgroupgettitle{glssymbols}}%
6230 \glshypernavigationsep
6231 \glsnavigationhyperlink{glsnumbers}{\glsgroupgettitle{glsnumbers}}%
6232 \glshypernavigationsep
6233 }

```

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.

```

6234 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]

```

inline Define the inline style.

```

6235 \newglossarystyle{inline}{%
    Start of glossary sets up first empty separator between entries. (This is then
    changed by \glossentry)
6236 \renewenvironment{theglossary}%
6237     {%
6238         \def\gls@inlinesep{}%
6239         \def\gls@inlinesubsep{}%
6240         \def\gls@inlinepostchild{}%
6241     }%
6242     {\glspostinline}%

```

No header:

```

6243 \renewcommand*{\glossaryheader}{}%
    No group headings (if heading is required, add \glsinlinedopostchild to
    start definition in case heading follows a child entry):
6244 \renewcommand*{\glsgroupheading}[1]{}%

```

Just display separator followed by name and description:

```

6245 \renewcommand{\glossentry}[2]{%
6246     \glsinlinedopostchild
6247     \gls@inlinesep
6248     \glsentryitem{##1}%
6249     \glsinlinenameformat{##1}{%
6250         \glossentryname{##1}%
6251     }%
6252     \ifglstdescsuppressed{##1}%
6253     {%
6254         \glsinlineemptydescformat
6255     }%
6256     \glossentrysymbol{##1}%
6257 }%
6258 {%
6259     ##2%
6260 }%

```

```

6261 }%
6262 {%
6263   \ifglshasdesc{##1}%
6264   {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
6265   {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
6266 }%
6267 \ifglshaschildren{##1}%
6268 {%
6269   \glsresetsubentrycounter
6270   \glsinlineparentchildseparator
6271   \def\gls@inlinesubsep{}%
6272   \def\gls@inlinepostchild{\glsinlinepostchild}%
6273 }%
6274 {}%
6275 \def\gls@inlinesep{\glsinlineseparator}%
6276 }%

```

Sub-entries display description:

```

6277 \renewcommand{\subglossentry}[3]{%
6278   \gls@inlinesubsep%
6279   \glsinlinesubnameformat{##2}{%
6280     \glossentryname{##2}}%
6281   \glssubentryitem{##2}%
6282   \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
6283   \def\gls@inlinesubsep{\glsinlinesubseparator}%
6284 }%

```

Nothing special between groups:

```

6285 \renewcommand*{\glsgroupskip}{}%
6286 }

```

\glsinlinedopostchild

```

6287 \newcommand*{\glsinlinedopostchild}{%
6288   \gls@inlinepostchild
6289   \def\gls@inlinepostchild{}%
6290 }

```

\glsinlineseparator Separator to use between entries.

```

6291 \newcommand*{\glsinlineseparator}{;\space}

```

\glsinlinesubseparator Separator to use between sub-entries.

```

6292 \newcommand*{\glsinlinesubseparator}{,\space}

```

\glsinlineparentchildseparator Separator to use between parent and children.

```

6293 \newcommand*{\glsinlineparentchildseparator}{:\space}

```

\glsinlinepostchild Hook to use between child and next entry

```

6294 \newcommand*{\glsinlinepostchild}{}

```

`\glspostinline` Terminator for inline glossary.
6295 `\newcommand*{\glspostinline}{\glspostdescription\space}`

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):
6296 `\newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}`

`\glsinlinedescformat` Formats the entry's description, symbol and location list:
6297 `\newcommand*{\glsinlinedescformat}[3]{\space#1}`

`\lineemptydescformat` Formats the entry's symbol and location list when the description is empty:
6298 `\newcommand*{\glsinlineemptydescformat}[2]{}`

`\inlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):
6299 `\newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}`

`\inlinesubdescformat` Formats the subentry's description, symbol and location list:
6300 `\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.

6301 `\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.

6302 `\newglossarystyle{list}{%`
Use description environment:
6303 `\renewenvironment{theglossary}{%`
6304 `{\begin{description}}{\end{description}}%`

No header at the start of the environment:
6305 `\renewcommand*{\glossaryheader}{}%`

No group headings:
6306 `\renewcommand*{\glsgroupheading}[1]{}%`

Main (level 0) entries start a new item in the list:
6307 `\renewcommand*{\glossentry}[2]{%`
6308 `\item[\glstentryitem{##1}]%`
6309 `\glstarget{##1}{\glossentryname{##1}}]`
6310 `\glossentrydesc{##1}\glspostdescription\space ##2}%`

Sub-entries continue on the same line:

```

6311 \renewcommand*{\subglossentry}[3]{%
6312   \glssubentryitem{##2}%
6313   \glstarget{##2}{\strut}%
6314   \glossentrydesc{##2}\glspostdescription\space ##3.}%
6315 %   \end{macrocode}
6316 % Add vertical space between groups:
6317 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
6318 %   \begin{macrocode}
6319 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
6320 }
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```

6321 \newglossarystyle{listgroup}{%
  Base it on the list style:
6322   \setglossarystyle{list}%
  Each group has a heading:
6323   \renewcommand*{\glsgroupheading}[1]{\item[\glsgrouptitle{##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.

```

6324 \newglossarystyle{listhypergroup}{%
  Base it on the list style:
6325   \setglossarystyle{list}%
  Add navigation links at the start of the environment:
6326   \renewcommand*{\glossaryheader}{%
6327     \item[\glsnavigation]}%
  Each group has a heading with a hypertarget:
6328   \renewcommand*{\glsgroupheading}[1]{%
6329     \item[\glsnavigationhypertarget{##1}{\glsgrouptitle{##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.

```

6330 \newglossarystyle{altlist}{%
  Base it on the list style:
6331   \setglossarystyle{list}%
  Main (level 0) entries start a new item in the list with a line break after the entry
  name:
6332   \renewcommand*{\glossentry}[2]{%
6333     \item[\glssubentryitem{##1}%
6334       \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.

```
6335      \mbox{}\par\nobreak\@afterheading
6336      \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
6337  \renewcommand{\subglossentry}[3]{%
6338    \par
6339    \glssubentryitem{##2}%
6340    \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
6341 }
```

`altlistgroup` The `altlistgroup` glossary style is like the `altlist` style, but the glossary groups have headings.

```
6342 \newglossarystyle{altlistgroup}{%
    Base it on the altlist style:
6343   \setglossarystyle{altlist}%
    Each group has a heading:
6344   \renewcommand*{\glsgroupheading}[1]{\item[\glsgrouptitle{##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.

```
6345 \newglossarystyle{altlisthypergroup}{%
    Base it on the altlist style:
6346   \setglossarystyle{altlist}%
    Add navigation links at the start of the environment:
6347   \renewcommand*{\glossaryheader}{%
6348     \item[\glsnavigation]}%
    Each group has a heading with a hypertarget:
6349   \renewcommand*{\glsgroupheading}[1]{%
6350     \item[\glsnavhypertarget{##1}]{\glsgrouptitle{##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.

```
6351 \newglossarystyle{listdotted}{%
    Base it on the list style:
6352   \setglossarystyle{list}%
```


Each main (level 0) entry starts a new item:

```
6353 \renewcommand*{\glossentry}[2]{%
6354   \item[\makebox[\glslistdottedwidth][l]{%
6355     \glentryitem{##1}%
6356     \glstarget{##1}{\glossentryname{##1}}}%
6357   \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
6358 \renewcommand*{\subglossentry}[3]{%
6359   \item[\makebox[\glslistdottedwidth][l]{%
6360     \glssubentryitem{##2}%
6361     \glstarget{##2}{\glossentryname{##2}}}%
6362   \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
6363 }
```

`\glslistdottedwidth`

```
6364 \newlength\glslistdottedwidth
6365 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
6366 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
6367 \setglossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
6368 \renewcommand*{\glossentry}[2]{%
6369   \item[\glentryitem{##1}\glstarget{##1}{\glossentryname{##1}}}%
6370 }
```

4.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the `longtable` environment in the glossary.

```
6371 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
6372 \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`.)

```
6373 \@ifundefined{glsdescwidth}{%
6374   \newlength\glsdescwidth
6375   \setlength{\glsdescwidth}{0.6\hsize}
6376 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
6377 \@ifundefined{glspagelistwidth}{%  
6378   \newlength{glspagelistwidth}  
6379   \setlength{glspagelistwidth}{0.1\hsize}  
6380 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
6381 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
6382   \renewenvironment{theglossary}{%  
6383     {\begin{longtable}{lp{\glstdescwidth}}}%  
6384     {\end{longtable}}}%
```

Do nothing at the start of the environment:

```
6385   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
6386   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
6387   \renewcommand{\glossentry}[2]{%  
6388     \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
6389     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline  
6390   }%
```

Sub entries displayed on the following row without the name:

```
6391   \renewcommand{\subglossentry}[3]{%  
6392     &  
6393     \glssubentryitem{##2}%  
6394     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space  
6395     ##3\tabularnewline  
6396   }%
```

Blank row between groups:

```
6397   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &  
6398   \tabularnewline\fi}%  
6399 }
```

`longborder` The longborder style is like the above, but with horizontal and vertical lines:

```
6400 \newglossarystyle{longborder}{%
```

Base it on the glostylelong style:

```
6401   \setglossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
6402   \renewenvironment{theglossary}{%  
6403     \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
6404   \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
6405 }
```

longheader The longheader style is like the long style but with a header:

```
6406 \newglossarystyle{longheader}{%
```

Base it on the `glostylelong` style:

```
6407 \setglossarystyle{long}%
```

Set the table's header:

```
6408 \renewcommand*{\glossaryheader}{%
```

```
6409 \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%
```

```
6410 }
```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```
6411 \newglossarystyle{longheaderborder}{%
```

Base it on the `glostylelongborder` style:

```
6412 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
6413 \renewcommand*{\glossaryheader}{%
```

```
6414 \hline\bfseries \entryname & \bfseries
```

```
6415 \descriptionname\tabularnewline\hline
```

```
6416 \endhead
```

```
6417 \hline\endfoot}%
```

```
6418 }
```

long3col The long3col style is like long but with 3 columns

```
6419 \newglossarystyle{long3col}{%
```

Use a longtable with 3 columns:

```
6420 \renewenvironment{theglossary}%
```

```
6421 {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
```

```
6422 {\end{longtable}}%
```

No table header:

```
6423 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
6424 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
6425 \renewcommand{\glossentry}[2]{%
```

```
6426 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
```

```
6427 \glossentrydesc{##1} & ##2\tabularnewline
```

```
6428 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
6429 \renewcommand{\subglossentry}[3]{%
```

```
6430 &
```

```
6431 \glssubentryitem{##2}%
```

```
6432 \glstarget{##2}{\strut}\glossentrydesc{##2} &
```

```

6433     ##3\tabularnewline
6434 }%

Blank row between groups:
6435 \renewcommand*{\glsgroupskip}{%
6436 \ifglsgroupskip\else & &\tabularnewline\fi}%
6437 }

```

long3colborder The long3colborder style is like the long3col style but with a border:

```

6438 \newglossarystyle{long3colborder}{%

Base it on the glostylelong3col style:
6439 \setglossarystyle{long3col}%

Use a longtable with 3 columns with vertical lines around them:
6440 \renewenvironment{theglossary}%
6441 {\begin{longtable}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
6442 {\end{longtable}}%

Place horizontal lines at the head and foot of the table:
6443 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6444 }

```

long3colheader The long3colheader style is like long3col but with a header row:

```

6445 \newglossarystyle{long3colheader}{%

Base it on the glostylelong3col style:
6446 \setglossarystyle{long3col}%

Set the table's header:
6447 \renewcommand*{\glossaryheader}{%
6448 \bfseries\entryname&\bfseries\descriptionname&
6449 \bfseries\pagelistname\tabularnewline\endhead}%
6450 }

```

long3colheaderborder The long3colheaderborder style is like the above but with a border

```

6451 \newglossarystyle{long3colheaderborder}{%

Base it on the glostylelong3colborder style:
6452 \setglossarystyle{long3colborder}%

Set the table's header and add horizontal line at table's foot:
6453 \renewcommand*{\glossaryheader}{%
6454 \hline
6455 \bfseries\entryname&\bfseries\descriptionname&
6456 \bfseries\pagelistname\tabularnewline\hline\endhead
6457 \hline\endfoot}%
6458 }

```

long4col The long4col style has four columns where the third column contains the value of the associated symbol key.

```

6459 \newglossarystyle{long4col}{%

```

Use a longtable with 4 columns:

```
6460 \renewenvironment{theglossary}%  
6461 {\begin{longtable}{llll}}%  
6462 {\end{longtable}}%
```

No table header:

```
6463 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6464 \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):

```
6465 \renewcommand{\glossentry}[2]{%  
6466 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
6467 \glossentrydesc{##1} &  
6468 \glossentrysymbol{##1} &  
6469 ##2\tabularnewline  
6470 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
6471 \renewcommand{\subglossentry}[3]{%  
6472 &  
6473 \glssubentryitem{##2}%  
6474 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
6475 \glossentrysymbol{##2} & ##3\tabularnewline  
6476 }%
```

Blank row between groups:

```
6477 \renewcommand*{\glsgroupskip}{}%  
6478 \ifglsgnognroupskip\else & & \tabularnewline\fi}%  
6479 }
```

long4colheader The long4colheader style is like long4col but with a header row.

```
6480 \newglossarystyle{long4colheader}{}%
```

Base it on the glostylelong4col style:

```
6481 \setglossarystyle{long4col}{}%
```

Table has a header:

```
6482 \renewcommand*{\glossaryheader}{}%  
6483 \bfseries\entryname&\bfseries\descriptionname&  
6484 \bfseries \symbolname&  
6485 \bfseries\pagelistname\tabularnewline\endhead}%  
6486 }
```

long4colborder The long4colborder style is like long4col but with a border.

```
6487 \newglossarystyle{long4colborder}{}%
```

Base it on the glostylelong4col style:

```
6488 \setglossarystyle{long4col}{}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
6489 \renewenvironment{theglossary}%
6490 {\begin{longtable}{|l|l|l|l|}}%
6491 {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
6492 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6493 }
```

long4colheaderborder The long4colheaderborder style is like the above but with a border.

```
6494 \newglossarystyle{long4colheaderborder}{%
```

Base it on the glostylelong4col style:

```
6495 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
6496 \renewenvironment{theglossary}%
6497 {\begin{longtable}{|l|l|l|l|}}%
6498 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
6499 \renewcommand*{\glossaryheader}{%
6500 \hline\bfseries\entryname&\bfseries\descriptionname&
6501 \bfseries \symbolname&
6502 \bfseries\pagelistname\tabularnewline\hline\endhead
6503 \hline\endfoot}%
6504 }
```

altlong4col The altlong4col style is like the long4col style but can have multiline descriptions and page lists.

```
6505 \newglossarystyle{altlong4col}{%
```

Base it on the glostylelong4col style:

```
6506 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6507 \renewenvironment{theglossary}%
6508 {\begin{longtable}{lp{\glstdescwidth}lp{\glspagelistwidth}}}%
6509 {\end{longtable}}%
6510 }
```

altlong4colheader The altlong4colheader style is like altlong4col but with a header row.

```
6511 \newglossarystyle{altlong4colheader}{%
```

Base it on the glostylelong4colheader style:

```
6512 \setglossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6513 \renewenvironment{theglossary}%
```

```

6514    {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
6515    {\end{longtable}}}%
6516 }

```

`altlong4colborder` The `altlong4colborder` style is like `altlong4col` but with a border.

```

6517 \newglossarystyle{altlong4colborder}{%

```

Base it on the `glostylelong4colborder` style:

```

6518    \setglossarystyle{long4colborder}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

6519    \renewenvironment{theglossary}%
6520    {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
6521    {\end{longtable}}}%
6522 }

```

`ong4colheaderborder` The `altlong4colheaderborder` style is like the above but with a header as well as a border.

```

6523 \newglossarystyle{altlong4colheaderborder}{%

```

Base it on the `glostylelong4colheaderborder` style:

```

6524    \setglossarystyle{long4colheaderborder}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

6525    \renewenvironment{theglossary}%
6526    {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
6527    {\end{longtable}}}%
6528 }

```

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.

```

6529 \ProvidesPackage{glossary-longragged}[2013/11/14 v4.0 (NLCT)]

```

Requires the package:

```

6530 \RequirePackage{array}

```

Requires the package:

```

6531 \RequirePackage{longtable}

```

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```

6532 \@ifundefined{glsdescwidth}{%
6533    \newlength\glsdescwidth
6534    \setlength{\glsdescwidth}{0.6\hsize}
6535 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
6536 \@ifundefined{glspagelistwidth}{%
6537   \newlength{glspagelistwidth
6538   \setlength{glspagelistwidth}{0.1\hsize}
6539 }{}
```

`longragged` The `longragged` glossary style is like the `long` but uses ragged right formatting for the description column.

```
6540 \newglossarystyle{longragged}{%
```

Use `longtable` with two columns:

```
6541   \renewenvironment{theglossary}{%
6542     \begin{longtable}{l>\raggedright}p{glstdescwidth}}}%
6543     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
6544   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
6545   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
6546   \renewcommand{\glossentry}[2]{%
6547     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6548     \glossentrydesc{##1}\glspostdescription\space ##2%
6549     \tabularnewline
6550   }%
```

Sub entries displayed on the following row without the name:

```
6551   \renewcommand{\subglossentry}[3]{%
6552     &
6553     \glssubentryitem{##2}%
6554     \glstarget{##2}{\strut}\glossentrydesc{##2}%
6555     \glspostdescription\space ##3%
6556     \tabularnewline
6557   }%
```

Blank row between groups:

```
6558   \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
6559 }
```

`longraggedborder` The `longraggedborder` style is like the above, but with horizontal and vertical lines:

```
6560 \newglossarystyle{longraggedborder}{%
```

Base it on the `glostylelongragged` style:

```
6561   \setglossarystyle{longragged}%
```

Use `longtable` with two columns with vertical lines between each column:

```
6562   \renewenvironment{theglossary}{%
6563     \begin{longtable}{|l|>\raggedright}p{glstdescwidth}|}%
6564     {\end{longtable}}%
```


Place horizontal lines at the head and foot of the table:

```
6565 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6566 }
```

`longraggedheader` The `longraggedheader` style is like the `longragged` style but with a header:

```
6567 \newglossarystyle{longraggedheader}{%
```

Base it on the `glostylelongragged` style:

```
6568 \setglossarystyle{longragged}%
```

Set the table's header:

```
6569 \renewcommand*{\glossaryheader}{%
6570 \bfseries \entryname & \bfseries \descriptionname
6571 \tabularnewline\endhead}%
6572 }
```

`longraggedheaderborder` The `longraggedheaderborder` style is like the `longragged` style but with a header and border:

```
6573 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
6574 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
6575 \renewcommand*{\glossaryheader}{%
6576 \hline\bfseries \entryname & \bfseries \descriptionname
6577 \tabularnewline\hline
6578 \endhead
6579 \hline\endfoot}%
6580 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
6581 \newglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns:

```
6582 \renewenvironment{theglossary}%
6583 {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
6584 >{\raggedright}p{\glspagelistwidth}}}%
6585 {\end{longtable}}%
```

No table header:

```
6586 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
6587 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
6588 \renewcommand{\glossentry}[2]{%
6589 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6590 \glossentrydesc{##1} & ##2\tabularnewline
6591 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
6592 \renewcommand{\subglossentry}[3]{%
6593     &
6594     \glssubentryitem{##2}%
6595     \glstarget{##2}{\strut}\glossentrydesc{##2} &
6596     ##3\tabularnewline
6597 }
```

Blank row between groups:

```
6598 \renewcommand*{\glsgroupskip}{%
6599     \ifglsgnogroupskip\else & &\tabularnewline\fi}%
6600 }
```

longragged3colborder The longragged3colborder style is like the longragged3col style but with a border:

```
6601 \newglossarystyle{longragged3colborder}{%
```

Base it on the glostylelongragged3col style:

```
6602 \setglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns with vertical lines around them:

```
6603 \renewenvironment{theglossary}{%
6604     {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|%
6605      >{\raggedright}p{\glspagelistwidth}|}%
6606     {\end{longtable}}}%
```

Place horizontal lines at the head and foot of the table:

```
6607 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6608 }
```

longragged3colheader The longragged3colheader style is like longragged3col but with a header row:

```
6609 \newglossarystyle{longragged3colheader}{%
```

Base it on the glostylelongragged3col style:

```
6610 \setglossarystyle{longragged3col}{%
```

Set the table's header:

```
6611 \renewcommand*{\glossaryheader}{%
6612     \bfseries\entryname&\bfseries\descriptionname&
6613     \bfseries\pagelistname\tabularnewline\endhead}%
6614 }
```

longragged3colheaderborder The longragged3colheaderborder style is like the above but with a border

```
6615 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the glostylelongragged3colborder style:

```
6616 \setglossarystyle{longragged3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
6617 \renewcommand*{\glossaryheader}{%
6618 \hline
6619 \bfseries\entryname&\bfseries\descriptionname&
6620 \bfseries\pagelistname\tabularnewline\hline\endhead
6621 \hline\endfoot}%
6622 }
```

`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.

```
6623 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6624 \renewenvironment{theglossary}%
6625 {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}l%
6626 >{\raggedright}p{\glspagelistwidth}}}%
6627 {\end{longtable}}}%
```

No table header:

```
6628 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6629 \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):

```
6630 \renewcommand{\glossentry}[2]{%
6631 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6632 \glossentrydesc{##1} & \glossentrydesc{##1} &
6633 ##2\tabularnewline
6634 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
6635 \renewcommand{\subglossentry}[3]{%
6636 &
6637 \glssubentryitem{##2}%
6638 \glstarget{##2}{\strut}\glossentrydesc{##2} &
6639 \glossentrysymbol{##2} & ##3\tabularnewline
6640 }%
```

Blank row between groups:

```
6641 \renewcommand*{\glsgroupskip}{%
6642 \ifglsnogroupskip\else & & \tabularnewline\fi}%
6643 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
6644 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
6645 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6646 \renewenvironment{theglossary}%
6647   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
6648     >{\raggedright}p{\glspagelistwidth}}}%
6649   {\end{longtable}}%
```

Table has a header:

```
6650 \renewcommand*{\glossaryheader}{%
6651   \bfseries\entryname&\bfseries\descriptionname&
6652   \bfseries \symbolname&
6653   \bfseries\pagelistname\tabularnewline\endhead}%
6654 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
6655 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
6656 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6657 \renewenvironment{theglossary}%
6658   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
6659     >{\raggedright}p{\glspagelistwidth}|}%
6660   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
6661 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6662 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
6663 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
6664 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6665 \renewenvironment{theglossary}%
6666   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
6667     >{\raggedright}p{\glspagelistwidth}|}%
6668   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
6669 \renewcommand*{\glossaryheader}{%
6670   \hline\bfseries\entryname&\bfseries\descriptionname&
```

```

6671 \bfseries \symbolname&
6672 \bfseries\pagelistname\tabularnewline\hline\endhead
6673 \hline\endfoot}%
6674 }

```

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.

```

6675 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]

```

Required packages:

```

6676 \RequirePackage{multicol}
6677 \RequirePackage{glossary-tree}

```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```

6678 \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 multicol, but the title isn't part of the glossary style.)

```

6679 \newglossarystyle{mcolindex}{%
6680 \setglossarystyle{index}%
6681 \renewenvironment{theglossary}%
6682 {%
6683 \begin{multicols}{\glsmcols}
6684 \setlength{\parindent}{0pt}%
6685 \setlength{\parskip}{0pt plus 0.3pt}%
6686 \let\item\@idxitem}%
6687 {\end{multicols}}%
6688 }

```

`mcolindexgroup` As `mcolindex` but has headings:

```

6689 \newglossarystyle{mcolindexgroup}{%
6690 \setglossarystyle{mcolindex}%
6691 \renewcommand*{\glsgroupheading}[1]{%
6692 \item\textbf{\glsgroupheading{##1}}\indexspace}%
6693 }

```

`mcolindexhypergroup` The `mcolindexhypergroup` style is like the `mcolindexgroup` style but has hyper navigation.

```

6694 \newglossarystyle{mcolindexhypergroup}{%

```

Base it on the `glostylemcolindex` style:

```

6695 \setglossarystyle{mcolindex}%

```

Put navigation links to the groups at the start of the glossary:

```

6696 \renewcommand*{\glossaryheader}{%
6697 \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.

```
6698 \renewcommand*{\glsgroupheading}[1]{%
6699 \item\textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
6700 \indexspace}%
6701 }
```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```
6702 \newglossarystyle{mcoltree}{%
6703 \setglossarystyle{tree}%
6704 \renewenvironment{theglossary}%
6705 {%
6706 \begin{multicols}{\glsmcols}
6707 \setlength{\parindent}{0pt}%
6708 \setlength{\parskip}{0pt plus 0.3pt}%
6709 }%
6710 {\end{multicols}}}%
6711 }
```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```
6712 \newglossarystyle{mcoltreegroup}{%
Base it on the glostylemcoltree style:
6713 \setglossarystyle{mcoltree}%
Each group has a heading (in bold) followed by a vertical gap):
6714 \renewcommand{\glsgroupheading}[1]{\par
6715 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
6716 }
```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
6717 \newglossarystyle{mcoltreehypergroup}{%
Base it on the glostylemcoltree style:
6718 \setglossarystyle{mcoltree}%
Put navigation links to the groups at the start of the theglossary environment:
6719 \renewcommand*{\glossaryheader}{%
6720 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
Each group has a heading (in bold with a target) followed by a vertical gap):
6721 \renewcommand*{\glsgroupheading}[1]{%
6722 \par\noindent
6723 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
6724 \indexspace}%
6725 }
```

mcoltreenoname Multi-column index style. Same as the **treenoname**, but puts the glossary in multiple columns.

```

6726 \newglossarystyle{mcoltreenoname}{%
6727   \setglossarystyle{treenoname}%
6728   \renewenvironment{theglossary}%
6729   {%

6730       \begin{multicols}{\glsmcols}
6731       \setlength{\parindent}{0pt}%
6732       \setlength{\parskip}{0pt plus 0.3pt}%
6733   }%
6734   {\end{multicols}}}%
6735 }
```

mcoltreenonamegroup Like the **mcoltreenoname** style but the glossary groups have headings.

```

6736 \newglossarystyle{mcoltreenonamegroup}{%

    Base it on the glostylemcoltreenoname style:
6737   \setglossarystyle{mcoltreenoname}%

    Give each group a heading:
6738   \renewcommand{\glsgroupheading}[1]{\par
6739     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
6740 }
```

treenonamehypergroup The **mcoltreenonamehypergroup** style is like the **mcoltreenonamegroup** style, but has a set of links to the groups at the start of the glossary.

```

6741 \newglossarystyle{mcoltreenonamehypergroup}{%

    Base it on the glostylemcoltreenoname style:
6742   \setglossarystyle{mcoltreenoname}%

    Put navigation links to the groups at the start of the theglossary environment:
6743   \renewcommand*{\glossaryheader}{%
6744     \par\noindent\textbf{\glsnavigation}\par\indexspace}%

    Each group has a heading (in bold with a target) followed by a vertical gap):
6745   \renewcommand*{\glsgroupheading}[1]{%
6746     \par\noindent
6747     \textbf{\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
6748     \indexspace}%
6749 }
```

mcolalmtree Multi-column index style. Same as the **almtree**, but puts the glossary in multiple columns.

```

6750 \newglossarystyle{mcolalmtree}{%
6751   \setglossarystyle{almtree}%
6752   \renewenvironment{theglossary}%
6753   {%
```

```

6754     \begin{multicols}{\glsmcols}
6755     \def\@gls@prevlevel{-1}%
6756     \mbox{}\par
6757 }%
6758 {\par\end{multicols}}%
6759 }

```

mcolalttreegroup Like the mcolalttree style but the glossary groups have headings.

```

6760 \newglossarystyle{mcolalttreegroup}{%
    Base it on the glostylemcolalttree style:
6761   \setglossarystyle{mcolalttree}%
    Give each group a heading.
6762   \renewcommand{\glsgroupheading}[1]{\par
6763     \def\@gls@prevlevel{-1}%
6764     \hangindent0pt\relax
6765     \parindent0pt\relax
6766     \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
6767 }

```

olalttreehypergroup The mcolalttreehypergroup style is like the mcolalttreegroup style, but has a set of links to the groups at the start of the glossary.

```

6768 \newglossarystyle{mcolalttreehypergroup}{%
    Base it on the glostylemcolalttree style:
6769   \setglossarystyle{mcolalttree}%
    Put the navigation links in the header
6770   \renewcommand*\glossaryheader{%
6771     \par
6772     \def\@gls@prevlevel{-1}%
6773     \hangindent0pt\relax
6774     \parindent0pt\relax
6775     \textbf{\glsnavigation}\par\indexspace}%
    Put a hypertarget at the start of each group
6776   \renewcommand*\glsgroupheading[1]{%
6777     \par
6778     \def\@gls@prevlevel{-1}%
6779     \hangindent0pt\relax
6780     \parindent0pt\relax
6781     \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
6782     \indexspace}}

```

4.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```

6783 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]

```


Requires the package:

```
6784 \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.

```
6785 \@ifundefined{glsdescwidth}{%
6786   \newlength{glsdescwidth
6787   \setlength{glsdescwidth}{0.6\hsize}
6788 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
6789 \@ifundefined{glspagelistwidth}{%
6790   \newlength{glspagelistwidth
6791   \setlength{glspagelistwidth}{0.1\hsize}
6792 }{}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
6793 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
6794   \renewenvironment{theglossary}%
6795   {\tablehead{}\tabletail}%
6796   \begin{supertabular}{lp{glsdescwidth}}%
6797   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
6798   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6799   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
6800   \renewcommand{\glossentry}[2]{%
6801     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6802     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
6803   }%
```

Sub entries put in a row (no name, description and page list in second column):

```
6804   \renewcommand{\subglossentry}[3]{%
6805     &
6806     \glssubentryitem{##2}%
6807     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
6808     ##3\tabularnewline
6809   }%
```

Blank row between groups:

```
6810 \renewcommand*{\glsgroupskip}{%  
6811 \ifglsnogroupskip\else & \tabularnewline\fi}%  
6812 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
6813 \newglossarystyle{superborder}{%
```

Base it on the glostylesuper style:

```
6814 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
6815 \renewenvironment{theglossary}%  
6816 {\tablehead{\hline}\tabletail{\hline}%  
6817 \begin{supertabular}{|l|p{\glsdescwidth}|}%  
6818 {\end{supertabular}}%  
6819 }
```

superheader The superheader style is like the super style, but with a header:

```
6820 \newglossarystyle{superheader}{%
```

Base it on the glostylesuper style:

```
6821 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
6822 \renewenvironment{theglossary}%  
6823 {\tablehead{\bfseries \entryname &  
6824 \bfseries \descriptionname \tabularnewline}%  
6825 \tabletail{}}%  
6826 \begin{supertabular}{|lp{\glsdescwidth}|}%  
6827 {\end{supertabular}}%  
6828 }
```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
6829 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
6830 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```
6831 \renewenvironment{theglossary}%  
6832 {\tablehead{\hline\bfseries \entryname &  
6833 \bfseries \descriptionname \tabularnewline\hline}%  
6834 \tabletail{\hline}  
6835 \begin{supertabular}{|l|p{\glsdescwidth}|}%  
6836 {\end{supertabular}}%  
6837 }
```

super3col The super3col style is like the super style, but with 3 columns:

```
6838 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```
6839 \renewenvironment{theglossary}%  
6840 {\tablehead{}\tabletail{}}%  
6841 \begin{supertabular}{\lp{\glstdescwidth}\p{\glspagelistwidth}}}%  
6842 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
6843 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6844 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
6845 \renewcommand{\glossentry}[2]{%  
6846 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
6847 \glossentrydesc{##1} & ##2\tabularnewline  
6848 }%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
6849 \renewcommand{\subglossentry}[3]{%  
6850 &  
6851 \glssubentryitem{##2}%  
6852 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
6853 ##3\tabularnewline  
6854 }%
```

Blank row between groups:

```
6855 \renewcommand*{\glsgroupskip}{%  
6856 \ifglsgroupskip\else & &\tabularnewline\fi}%  
6857 }
```

super3colborder The super3colborder style is like the super3col style, but with a border:

```
6858 \newglossarystyle{super3colborder}{%
```

Base it on the glostylessuper3col style:

```
6859 \setglossarystyle{super3col}%
```

Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:

```
6860 \renewenvironment{theglossary}%  
6861 {\tablehead{\hline}\tabletail{\hline}}%  
6862 \begin{supertabular}{\lp{\glstdescwidth}\p{\glspagelistwidth}}}%  
6863 {\end{supertabular}}%  
6864 }
```

super3colheader The **super3colheader** style is like the **super3col** style but with a header row:

```
6865 \newglossarystyle{super3colheader}{%
```

Base it on the **glostylesuper3col** style:

```
6866 \setglossarystyle{super3col}{%
```

Put the glossary in a **supertabular** environment with three columns, a header and no tail:

```
6867 \renewenvironment{theglossary}%
6868 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
6869 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
6870 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
6871 {\end{supertabular}}}%
6872 }
```

super3colheaderborder The **super3colheaderborder** style is like the **super3col** style but with a header and border:

```
6873 \newglossarystyle{super3colheaderborder}{%
```

Base it on the **glostylesuper3colborder** style:

```
6874 \setglossarystyle{super3colborder}{%
```

Put the glossary in a **supertabular** environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
6875 \renewenvironment{theglossary}%
6876 {\tablehead{\hline
6877 \bfseries\entryname&\bfseries\descriptionname&
6878 \bfseries\pagelistname\tabularnewline\hline}%
6879 \tabletail{\hline}%
6880 \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
6881 {\end{supertabular}}}%
6882 }
```

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.

```
6883 \newglossarystyle{super4col}{%
```

Put the glossary in a **supertabular** environment with four columns and no head or tail:

```
6884 \renewenvironment{theglossary}%
6885 {\tablehead{}\tabletail{}}%
6886 \begin{supertabular}{llll}}}%
6887 \end{supertabular}}%
```

Do nothing at the start of the table:

```
6888 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6889 \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:

```
6890 \renewcommand{\glossentry}[2]{%
6891   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6892   \glossentrydesc{##1} &
6893   \glossentrysymbol{##1} & ##3\tabularnewline
6894 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
6895 \renewcommand{\subglossentry}[3]{%
6896   &
6897   \glssubentryitem{##2}%
6898   \glstarget{##2}{\strut}\glossentrydesc{##2} &
6899   \glossentrysymbol{##2} & ##3\tabularnewline
6900 }%
```

Blank row between groups:

```
6901 \renewcommand*{\glsgroupskip}{%
6902   \ifglsnogroupskip\else & & \tabularnewline\fi}%
6903 }
```

super4colheader The super4colheader style is like the super4col but with a header row.

```
6904 \newglossarystyle{super4colheader}{%
```

Base it on the glostylesuper4col style:

```
6905 \setglossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
6906 \renewenvironment{theglossary}%
6907   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
6908     \bfseries\symbolname &
6909     \bfseries\pagelistname\tabularnewline}%
6910   \tabletail{}}%
6911   \begin{supertabular}{|l|l|l|l|}%
6912   {\end{supertabular}}%
6913 }
```

super4colborder The super4colborder style is like the super4col but with a border.

```
6914 \newglossarystyle{super4colborder}{%
```

Base it on the glostylesuper4col style:

```
6915 \setglossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```
6916 \renewenvironment{theglossary}%
6917   {\tablehead{\hline}\tabletail{\hline}%
6918   \begin{supertabular}{|l|l|l|l|}%
6919   {\end{supertabular}}%
6920 }
```

`per4colheaderborder` The `super4colheaderborder` style is like the `super4col` but with a header and border.

```
6921 \newglossarystyle{super4colheaderborder}{%
```

Base it on the `glostylesuper4col` style:

```
6922 \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:

```
6923 \renewenvironment{theglossary}%
6924 {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
6925 \bfseries\symbolname &
6926 \bfseries\pagelistname\tabularnewline\hline}%
6927 \tabletail{\hline}%
6928 \begin{supertabular}{|l|l|l|l|}%
6929 {\end{supertabular}}%
6930 }
```

`altsuper4col` The `altsuper4col` glossary style is like `super4col` but has provision for multiline descriptions.

```
6931 \newglossarystyle{altsuper4col}{%
```

Base it on the `glostylesuper4col` style:

```
6932 \setglossarystyle{super4col}%
```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```
6933 \renewenvironment{theglossary}%
6934 {\tablehead{}\tabletail}%
6935 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}%
6936 {\end{supertabular}}%
6937 }
```

`altsuper4colheader` The `altsuper4colheader` style is like the `altsuper4col` but with a header row.

```
6938 \newglossarystyle{altsuper4colheader}{%
```

Base it on the `glostylesuper4colheader` style:

```
6939 \setglossarystyle{super4colheader}%
```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```
6940 \renewenvironment{theglossary}%
6941 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
6942 \bfseries\symbolname &
6943 \bfseries\pagelistname\tabularnewline}\tabletail}%
6944 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}%
6945 {\end{supertabular}}%
6946 }
```

`altsuper4colborder` The `altsuper4colborder` style is like the `altsuper4col` but with a border.

```
6947 \newglossarystyle{altsuper4colborder}{%
```

Base it on the `glostylesuper4colborder` style:

```
6948 \setglossarystyle{super4colborder}%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
6949 \renewenvironment{theglossary}%
6950   {\tablehead{\hline}\tabletail{\hline}%
6951    \begin{supertabular}%
6952      {\lllp{\glstdescwidth}\lllp{\glspagelistwidth}}}%
6953   {\end{supertabular}}%
6954 }
```

`per4colheaderborder` The `altsuper4colheaderborder` style is like the `altsuper4col` but with a header and border.

```
6955 \newglossarystyle{altsuper4colheaderborder}{%
```

Base it on the `glostylesuper4colheaderborder` style:

```
6956 \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:

```
6957 \renewenvironment{theglossary}%
6958   {\tablehead{\hline
6959     \bfseries\entryname &
6960     \bfseries\descriptionname &
6961     \bfseries\symbolname &
6962     \bfseries\pagelistname\tabularnewline\hline}%
6963    \tabletail{\hline}%
6964    \begin{supertabular}%
6965      {\lllp{\glstdescwidth}\lllp{\glspagelistwidth}}}%
6966   {\end{supertabular}}%
6967 }
```

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.

```
6968 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
6969 \RequirePackage{array}
```

Requires the package:

```
6970 \RequirePackage{supertabular}
```

`\glstdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```

6971 \@ifundefined{glsdescwidth}{%
6972   \newlength{glsdescwidth
6973   \setlength{glsdescwidth}{0.6\hsize}
6974 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

6975 \@ifundefined{glspagelistwidth}{%
6976   \newlength{glspagelistwidth
6977   \setlength{glspagelistwidth}{0.1\hsize}
6978 }{}

```

`superragged` The superragged glossary style uses the supertabular environment.

```

6979 \newglossarystyle{superragged}{%

```

Put the glossary in a supertabular environment with two columns and no head or tail:

```

6980   \renewenvironment{theglossary}%
6981     {\tablehead{}\tabletail}%
6982     \begin{supertabular}{1>\raggedright}p{glsdescwidth}}%
6983     {\end{supertabular}}%

```

Do nothing at the start of the table:

```

6984   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

6985   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```

6986   \renewcommand{\glossentry}[2]{%
6987     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6988     \glossentrydesc{##1}\glspostdescription\space ##2%
6989     \tabularnewline
6990   }%

```

Sub entries put in a row (no name, description and page list in second column):

```

6991   \renewcommand{\subglossentry}[3]{%
6992     &
6993     \glssubentryitem{##2}%
6994     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
6995     ##3%
6996     \tabularnewline
6997   }%

```

Blank row between groups:

```

6998   \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else & \tabularnewline\fi}%
6999 }

```

`superraggedborder` The superraggedborder style is like the above, but with horizontal and vertical lines:

```

7000 \newglossarystyle{superraggedborder}{%

```


Base it on the `glostylessuperragged` style:

```
7001 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
7002 \renewenvironment{theglossary}%  
7003   {\tablehead{\hline}\tabletail{\hline}%  
7004    \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}}}%  
7005   {\end{supertabular}}%  
7006 }
```

`superraggedheader` The `superraggedheader` style is like the `super` style, but with a header:

```
7007 \newglossarystyle{superraggedheader}{%
```

Base it on the `glostylessuperragged` style:

```
7008 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
7009 \renewenvironment{theglossary}%  
7010   {\tablehead{\bfseries \entryname & \bfseries \descriptionname  
7011    \tabularnewline}%  
7012    \tabletail{}}%  
7013    \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}}}%  
7014   {\end{supertabular}}%  
7015 }
```

`superraggedheaderborder` The `superraggedheaderborder` style is like the `superragged` style but with a header and border:

```
7016 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the `glostylessuper` style:

```
7017 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
7018 \renewenvironment{theglossary}%  
7019   {\tablehead{\hline\bfseries \entryname &  
7020    \bfseries \descriptionname\tabularnewline\hline}%  
7021    \tabletail{\hline}  
7022    \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}}}%  
7023   {\end{supertabular}}%  
7024 }
```

`superragged3col` The `superragged3col` style is like the `superragged` style, but with 3 columns:

```
7025 \newglossarystyle{superragged3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
7026 \renewenvironment{theglossary}%
```

```

7027   {\tablehead{}\tabletail{}}%
7028   \begin{supertabular}{l>{\raggedright}p{\glstdescwidth}%
7029     >{\raggedright}p{\glspagelistwidth}}}%
7030   {\end{supertabular}}}%

```

Do nothing at the start of the table:

```

7031   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

7032   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7033   \renewcommand{\glossentry}[2]{%
7034     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7035     \glossentrydesc{##1} &
7036     ##2\tabularnewline
7037   }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

7038   \renewcommand{\subglossentry}[3]{%
7039     &
7040     \glssubentryitem{##2}%
7041     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7042     ##3\tabularnewline
7043   }%

```

Blank row between groups:

```

7044   \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
7045 }

```

superragged3colborder The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```

7046 \newglossarystyle{superragged3colborder}{%

```

Base it on the `glostylesuperragged3col` style:

```

7047   \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```

7048   \renewenvironment{theglossary}%
7049     {\tablehead{\hline}\tabletail{\hline}%
7050     \begin{supertabular}{|l|>{\raggedright}p{\glstdescwidth}|%
7051       >{\raggedright}p{\glspagelistwidth}|}%
7052     {\end{supertabular}}}%
7053 }

```

superragged3colheader The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```

7054 \newglossarystyle{superragged3colheader}{%

```

Base it on the `glostylesuperragged3col` style:

```
7055 \setglossarystyle{superragged3col}%
```

Put the glossary in a supertabular environment with three columns, a header and no tail:

```
7056 \renewenvironment{theglossary}%  
7057   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
7058     \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
7059   \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%  
7060     >{\raggedright}p{\glspagelistwidth}}}%  
7061   {\end{supertabular}}%  
7062 }
```

`ght3colheaderborder` The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```
7063 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the `glostylesuperragged3colborder` style:

```
7064 \setglossarystyle{superragged3colborder}%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
7065 \renewenvironment{theglossary}%  
7066   {\tablehead{\hline  
7067     \bfseries\entryname&\bfseries\descriptionname&  
7068     \bfseries\pagelistname\tabularnewline\hline}%  
7069   \tabletail{\hline}%  
7070   \begin{supertabular}{|1|>{\raggedright}p{\glsdescwidth}|%  
7071     >{\raggedright}p{\glspagelistwidth}|}%  
7072   {\end{supertabular}}%  
7073 }
```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```
7074 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
7075 \renewenvironment{theglossary}%  
7076   {\tablehead{}\tabletail{}}%  
7077   \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%  
7078     >{\raggedright}p{\glspagelistwidth}}}%  
7079   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7080 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7081 \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:

```

7082 \renewcommand{\glossentry}[2]{%
7083   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7084   \glossentrydesc{##1} &
7085   \glossentrysymbol{##1} & ##2\tabularnewline
7086 }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

7087 \renewcommand{\subglossentry}[3]{%
7088   &
7089   \glssubentryitem{##2}%
7090   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7091   \glossentrysymbol{##2} & ##3\tabularnewline
7092 }%

```

Blank row between groups:

```

7093 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & & \tabularnewline\fi}%
7094 }

```

`perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```

7095 \newglossarystyle{altsuperragged4colheader}{%

```

Base it on the `glostylealtsuperragged4col` style:

```

7096 \setglossarystyle{altsuperragged4col}%

```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

7097 \renewenvironment{theglossary}%
7098   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7099     \bfseries\symbolname &
7100     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7101   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
7102     >{\raggedright}p{\glspagelistwidth}}}%
7103   {\end{supertabular}}%
7104 }

```

`perragged4colborder` The `altsuperragged4colborder` style is like the `altsuperragged4col` style but with a border.

```

7105 \newglossarystyle{altsuperragged4colborder}{%

```

Base it on the `glostylealtsuperragged4col` style:

```

7106 \setglossarystyle{altsuper4col}%

```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

7107 \renewenvironment{theglossary}%
7108   {\tablehead{\hline}\tabletail{\hline}%

```

```

7109     \begin{supertabular}%
7110         {|l|>{\raggedright}p{\glsdescwidth}|l|}%
7111         >{\raggedright}p{\glspagelistwidth}|}}%
7112     {\end{supertabular}}%
7113 }

```

ged4colheaderborder The `altsuperragged4colheaderborder` style is like the `altsuperragged4col` style but with a header and border.

```

7114 \newglossarystyle{altsuperragged4colheaderborder}{%

```

Base it on the `glostylealtsuperragged4col` style:

```

7115 \setglossarystyle{altsuperragged4col}%

```

Put the glossary in a `supertabular` environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```

7116 \renewenvironment{theglossary}%
7117     {\tablehead{\hline
7118         \bfseries\entryname &
7119         \bfseries\descriptionname &
7120         \bfseries\symbolname &
7121         \bfseries\pagelistname\tabularnewline\hline}%
7122     \tabletail{\hline}%
7123     \begin{supertabular}%
7124         {|l|>{\raggedright}p{\glsdescwidth}|l|}%
7125         >{\raggedright}p{\glspagelistwidth}|}}%
7126     {\end{supertabular}}%
7127 }

```

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.

```

7128 \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.

```

7129 \newglossarystyle{index}{%

```

Set the paragraph indentation and skip and define `\item` to be the same as that used by `theindex`:

```

7130 \renewenvironment{theglossary}%
7131     {\setlength{\parindent}{0pt}%
7132     \setlength{\parskip}{0pt plus 0.3pt}%
7133     \let\item\@idxitem}%
7134     {\par}%

```

Do nothing at the start of the environment:

```
7135 \renewcommand*{\glossaryheader}{}%
```

No group headers:

```
7136 \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.

```
7137 \renewcommand*{\glossentry}[2]{%
7138   \item\glsentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7139   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7140   \space \glossentrydesc{##1}\glspostdescription\space ##2%
7141 }%
```

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.

```
7142 \renewcommand{\subglossentry}[3]{%
7143   \ifcase##1\relax
7144   % level 0
7145   \item
7146   \or
7147   % level 1
7148   \subitem
7149   \glssubentryitem{##2}%
7150   \else
7151   % all other levels
7152   \subsubitem
7153   \fi
7154   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7155   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7156   \space\glossentrydesc{##2}\glspostdescription\space ##3%
7157 }%
```

Vertical gap between groups is the same as that used by indices:

```
7158 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

`indexgroup` The `indexgroup` style is like the `index` style but has headings.

```
7159 \newglossarystyle{indexgroup}{%
```

Base it on the `glostyleindex` style:

```
7160 \setglossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```
7161 \renewcommand*{\glsgroupheading}[1]{%
7162   \item\textbf{\glsgetgrouptitle{##1}}\indexspace}%
7163 }
```

`indexhypergroup` The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```
7164 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```
7165 \setglossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```
7166 \renewcommand*{\glossaryheader}{%
```

```
7167 \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.

```
7168 \renewcommand*{\glsgroupheading}[1]{%
```

```
7169 \item\textbf{\glsnavhypertarget{##1}}{\glsgetgrouptitle{##1}}}%
```

```
7170 \indexspace}%
```

```
7171 }
```

`tree` The `tree` glossary style is similar in style to the `index` style, but can have arbitrary levels.

```
7172 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
7173 \renewenvironment{theglossary}%
```

```
7174 {\setlength{\parindent}{0pt}%
```

```
7175 \setlength{\parskip}{0pt plus 0.3pt}}%
```

```
7176 {}%
```

Do nothing at the start of the `theglossary` environment:

```
7177 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7178 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
7179 \renewcommand{\glossentry}[2]{%
```

```
7180 \hangindent0pt\relax
```

```
7181 \parindent0pt\relax
```

```
7182 \glsentryitem{##1}\textbf{\glstarget{##1}}{\glossentryname{##1}}}%
```

```
7183 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
```

```
7184 \space\glossentrydesc{##1}\glspostdescription\space##2\par
```

```
7185 }%
```

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.

```
7186 \renewcommand{\subglossentry}[3]{%
```

```
7187 \hangindent##1\glstreeindent\relax
```

```
7188 \parindent##1\glstreeindent\relax
```

```
7189 \ifnum##1=1\relax
```

```
7190 \glssubentryitem{##2}%
```

```

7191 \fi
7192 \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7193 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7194 \space\glossentrydesc{##2}\glspostdescription\space ##3\par
7195 }%

```

Vertical gap between groups is the same as that used by indices:

```

7196 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

treegroup Like the tree style but the glossary groups have headings.

```

7197 \newglossarystyle{treegroup}{%

```

Base it on the glostyletree style:

```

7198 \setglossarystyle{tree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```

7199 \renewcommand{\glsgroupheading}[1]{\par
7200 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7201 }

```

treehypergroup The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```

7202 \newglossarystyle{treehypergroup}{%

```

Base it on the glostyletree style:

```

7203 \setglossarystyle{tree}%

```

Put navigation links to the groups at the start of the theglossary environment:

```

7204 \renewcommand*{\glossaryheader}{%
7205 \par\noindent\textbf{\glsnavigation}\par\indexspace}%

```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

7206 \renewcommand*{\glsgroupheading}[1]{%
7207 \par\noindent
7208 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7209 \indexspace}%
7210 }

```

\glstreeindent Length governing left indent for each level of the tree style.

```

7211 \newlength\glstreeindent
7212 \setlength{\glstreeindent}{10pt}

```

treenoname The treenoname glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```

7213 \newglossarystyle{treenoname}{%

```

Set the paragraph indentation and skip:

```

7214 \renewenvironment{theglossary}%
7215 {\setlength{\parindent}{0pt}%
7216 \setlength{\parskip}{0pt plus 0.3pt}}%
7217 {}%

```


No header:

```
7218 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7219 \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.

```
7220 \renewcommand{\glossentry}[2]{%
7221   \hangindent0pt\relax
7222   \parindent0pt\relax
7223   \glstryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7224   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7225   \space\glossentrydesc{##1}\glspostdescription\space##2\par
7226 }%
```

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.

```
7227 \renewcommand{\subglossentry}[3]{%
7228   \hangindent##1\glstreeindent\relax
7229   \parindent##1\glstreeindent\relax
7230   \ifnum##1=1\relax
7231     \glssubentryitem{##2}%
7232   \fi
7233   \glstarget{##2}{\strut}%
7234   \glossentrydesc{##2}\glspostdescription\space##3\par
7235 }%
```

Vertical gap between groups is the same as that used by indices:

```
7236 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7237 }
```

treenonamegroup Like the `treenoname` style but the glossary groups have headings.

```
7238 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
7239 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
7240 \renewcommand{\glsgroupheading}[1]{\par
7241   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7242 }
```

treenonamehypergroup The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7243 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
7244 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
7245 \renewcommand*{\glossaryheader}{%
7246 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7247 \renewcommand*{\glsgroupheading}[1]{%
7248 \par\noindent
7249 \textbf{\glsnavhypertarget{##1}{\glsgroupheading{##1}}}\par
7250 \indexspace}%
7251 }
```

`\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.

```
7252 \newcommand*{\glsssetwidest}[2][0]{%
7253 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
7254 #2}%
7255 }
```

`\@glswidestname` Initialise `\@glswidestname`.

```
7256 \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`.

```
7257 \newglossarystyle{alttree}{%
```

Redefine theglossary environment.

```
7258 \renewenvironment{theglossary}%
7259 {\def\@gls@prevlevel{-1}%
7260 \mbox{}\par}%
7261 {\par}%
```

Set the header and group headers to nothing.

```
7262 \renewcommand*{\glossaryheader}{}%
7263 \renewcommand*{\glsgroupheading}[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
7264 \renewcommand{\glossentry}[2]{%
```

If the level hasn't changed, keep the same settings, otherwise change `\glstreeindent` accordingly.

```
7265 \ifnum\@gls@prevlevel=0\relax
7266 \else
```

Find out how big the indentation should be by measuring the widest entry.

```
7267 \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
```

Set the hangindent and paragraph indent.

```
7268 \hangindent\glstreeindent
7269 \parindent\glstreeindent
7270 \fi
```

Put the name to the left of the paragraph block.

```
7271 \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
7272 \glstryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
7273 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
```

Do the description followed by the description terminator and location list.

```
7274 \glossentrydesc{##1}\glspostdescription \space ##2\par
```

Set the previous level to 0.

```
7275 \def\@gls@prevlevel{0}%
7276 }%
```

Redefine the way sub-entries are displayed.

```
7277 \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.

```
7278 \ifnum##1=1\relax
7279 \glssubentryitem{##2}%
7280 \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```
7281 \ifnum\@gls@prevlevel=##1\relax
7282 \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in \gls@tmplen

```
7283 \@ifundefined{@glswidestname\romannumeral##1}{%
7284 \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}}%
7285 \settowidth{\gls@tmplen}{\textbf{%
7286 \csname @glswidestname\romannumeral##1\endcsname\space}}}%
```

Determine if going up or down a level

```
7287 \ifnum\@gls@prevlevel<##1\relax
```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```
7288 \setlength\glstreeindent\gls@tmplen
7289 \addtolength\glstreeindent\parindent
7290 \parindent\glstreeindent
7291 \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.

```
7292 \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
7293 \settowidth{\glstreeindent}{\textbf{%
7294 \@glswidestname\space}}}%
7295 \settowidth{\glstreeindent}{\textbf{%
7296 \csname @glswidestname\romannumeral\@gls@prevlevel
7297 \endcsname\space}}}%
```

Subtract this length from the previous level's paragraph indent and set to
`\glstreeindent`.

```
7298      \addtolength\parindent{-\glstreeindent}%
7299      \setlength\glstreeindent\parindent
7300      \fi
7301      \fi
```

Set the hanging indentation.

```
7302      \hangindent\glstreeindent
```

Put the name to the left of the paragraph block

```
7303      \makebox[0pt][r]{\makebox[\glstmplen][l]{%
7304      \textbf{\glstarget{##2}{\glossentryname{##2}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
7305      \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
```

Do the description followed by the description terminator and location list.

```
7306      \glossentrydesc{##2}\glspostdescription\space ##3\par
```

Set the previous level macro to the current level.

```
7307      \def\@gls@prevlevel{##1}%
7308      }%
```

Vertical gap between groups is the same as that used by indices:

```
7309      \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7310 }
```

almtreegroup Like the almtree style but the glossary groups have headings.

```
7311 \newglossarystyle{almtreegroup}{%
```

Base it on the glosstylealmtree style:

```
7312 \setglossarystyle{almtree}%
```

Give each group a heading.

```
7313 \renewcommand{\glsgroupheading}[1]{\par
7314 \def\@gls@prevlevel{-1}%
7315 \hangindent0pt\relax
7316 \parindent0pt\relax
7317 \textbf{\glsgrouptitle{##1}}\par\indexspace}%
7318 }
```

almtreehypergroup The almtreehypergroup style is like the almtreegroup style, but has a set of links to the groups at the start of the glossary.

```
7319 \newglossarystyle{almtreehypergroup}{%
```

Base it on the glosstylealmtree style:

```
7320 \setglossarystyle{almtree}%
```

Put the navigation links in the header

```
7321 \renewcommand*{\glossaryheader}{%
7322 \par
```

```

7323 \def\@gls@prevlevel{-1}%
7324 \hangindent0pt\relax
7325 \parindent0pt\relax
7326 \textbf{\glsnavigation}\par\indexspace}%

Put a hypertarget at the start of each group
7327 \renewcommand*\glsgroupheading[1]{%
7328 \par
7329 \def\@gls@prevlevel{-1}%
7330 \hangindent0pt\relax
7331 \parindent0pt\relax
7332 \textbf{\glsnavhypertarget{##1}\glsgetgrouptitle{##1}}\par
7333 \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.

```

7334 \NeedsTeXFormat{LaTeX2e}
7335 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

7336 \ifglsxindy
7337 \renewcommand*\GlsAddXdyAttribute[1]{%
7338 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
7339 \expandafter\toks@\expandafter{\@xdylocref}%
7340 \edef\@xdylocref{\the\toks@ ^^J%
7341 (markup-locref
7342 :open \string"\string~n\string\setentrycounter
7343 {\noexpand\glscounter}%
7344 \expandafter\string\csname#1\endcsname
7345 \expandafter\@gobble\string\{\string" ^^J
7346 :close \string"\expandafter\@gobble\string\}\string" ^^J
7347 :attr \string"#1\string"))}

```

Only has an effect before `\writeist`:

```

7348 \fi

```

`\GlsAddXdyCounters`

```

7349 \renewcommand*\GlsAddXdyCounters[1]{%
7350 \GlossariesWarning{\string\GlsAddXdyCounters\space not available
7351 in compatibility mode.}%
7352 }

```

Add predefined attributes

```

7353 \GlsAddXdyAttribute{glsnumberformat}
7354 \GlsAddXdyAttribute{textrm}

```

```

7355 \GlsAddXdyAttribute{textsf}
7356 \GlsAddXdyAttribute{texttt}
7357 \GlsAddXdyAttribute{textbf}
7358 \GlsAddXdyAttribute{textmd}
7359 \GlsAddXdyAttribute{textit}
7360 \GlsAddXdyAttribute{textup}
7361 \GlsAddXdyAttribute{textsl}
7362 \GlsAddXdyAttribute{textsc}
7363 \GlsAddXdyAttribute{emph}
7364 \GlsAddXdyAttribute{glshypernumber}
7365 \GlsAddXdyAttribute{hyper rm}
7366 \GlsAddXdyAttribute{hyper sf}
7367 \GlsAddXdyAttribute{hyper tt}
7368 \GlsAddXdyAttribute{hyper bf}
7369 \GlsAddXdyAttribute{hyper md}
7370 \GlsAddXdyAttribute{hyper it}
7371 \GlsAddXdyAttribute{hyper up}
7372 \GlsAddXdyAttribute{hyper sl}
7373 \GlsAddXdyAttribute{hyper sc}
7374 \GlsAddXdyAttribute{hyper emph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```

7375 \ifglxsindy
7376 \renewcommand*{\GlsAddXdyLocation}[2]{%
7377 \edef\@xdyuserlocationdefs{%
7378 \@xdyuserlocationdefs ^^J%
7379 (define-location-class \string"#1\string"^^J\space\space
7380 \space(#2))
7381 }%
7382 \edef\@xdyuserlocationnames{%
7383 \@xdyuserlocationnames^^J\space\space\space
7384 \string"#1\string"}%
7385 }
7386 \fi

```

\@do@wrglossary

```

7387 \renewcommand{\@do@wrglossary}[1]{%

```

Determine whether to use xindy or makeindex syntax

```

7388 \ifglxsindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

7389 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
7390 \def\@glo@range{}%
7391 \expandafter\if\@glo@prefix(\relax
7392 \def\@glo@range{:open-range}%
7393 \else
7394 \expandafter\if\@glo@prefix)\relax
7395 \def\@glo@range{:close-range}%

```

7396 \fi

7397 \fi

Get the location and escape any special characters

7398 \protected@edef\@glslocref{\theglsentrycounter}%

7399 \@gls@checkmkidxchars\@glslocref

Write to the glossary file using xindy syntax.

7400 \glossary[\csname glo@#1@type\endcsname]{%

7401 (indexentry :tkey (\csname glo@#1@index\endcsname)

7402 :locref \string"\@glslocref\string" %

7403 :attr \string"\@glo@suffix\string" \@glo@range

7404)

7405 }%

7406 \else

Convert the format information into the format required for makeindex

7407 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

Write to the glossary file using makeindex syntax.

7408 \glossary[\csname glo@#1@type\endcsname]{%

7409 \string\glossaryentry{\csname glo@#1@index\endcsname

7410 \@gls@encapchar\@glo@numfmt}{\theglsentrycounter}}%

7411 \fi

7412 }

\@set@glo@numformat Only had 3 arguments in v2.07

7413 \def\@set@glo@numformat#1#2#3{%

7414 \expandafter\@glo@check@mkidxrangechar#3\@nil

7415 \protected@edef#1{%

7416 \@glo@prefix setentrycounter[] {#2}%

7417 \expandafter\string\csname\@glo@suffix\endcsname

7418 }%

7419 \@gls@checkmkidxchars#1%

7420 }

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to
\glswrite.

7421 \ifglsxindy

7422 \def\writeist{%

7423 \openout\glswrite=\istfilename

7424 \write\glswrite{;; xindy style file created by the glossaries

7425 package in compatible-2.07 mode}%

7426 \write\glswrite{;; for document '\jobname' on

7427 \the\year-\the\month-\the\day}%

7428 \write\glswrite{^^J; required styles^^J}

7429 \@for\@xdystyle:=\@xdyrequiredstyles\do{%

7430 \ifx\@xdystyle\@empty

7431 \else

7432 \protected@write\glswrite{{(require

7433 \string"\@xdystyle.xdy\string")}}%

```

7434     \fi
7435 }%
7436 \write\glswrite{^^J%
7437     ; list of allowed attributes (number formats)^^J}%
7438 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
7439 \write\glswrite{^^J; user defined alphabets^^J}%
7440 \write\glswrite{\@xdyuseralphabets}%
7441 \write\glswrite{^^J; location class definitions^^J}%
7442 \protected@edef\@gls@roman{\@roman{0}\string"
7443     \string"roman-numbers-lowercase\string" :sep \string"}}%
7444 \@onelevel@sanitize\@gls@roman
7445 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
7446     :sep \string"}%
7447 \@onelevel@sanitize\@tmp
7448 \ifx\@tmp\@gls@roman
7449     \write\glswrite{(define-location-class
7450         \string"roman-page-numbers\string"^^J\space\space\space
7451         (\string"roman-numbers-lowercase\string")
7452         :min-range-length \@glsminrange)}}%
7453 \else
7454     \write\glswrite{(define-location-class
7455         \string"roman-page-numbers\string"^^J\space\space\space
7456         (:sep "\@gls@roman")
7457         :min-range-length \@glsminrange)}}%
7458 \fi
7459 \write\glswrite{(define-location-class
7460     \string"Roman-page-numbers\string"^^J\space\space\space
7461     (\string"roman-numbers-uppercase\string")
7462     :min-range-length \@glsminrange)}}%
7463 \write\glswrite{(define-location-class
7464     \string"arabic-page-numbers\string"^^J\space\space\space
7465     (\string"arabic-numbers\string")
7466     :min-range-length \@glsminrange)}}%
7467 \write\glswrite{(define-location-class
7468     \string"alpha-page-numbers\string"^^J\space\space\space
7469     (\string"alpha\string")
7470     :min-range-length \@glsminrange)}}%
7471 \write\glswrite{(define-location-class
7472     \string"Alpha-page-numbers\string"^^J\space\space\space
7473     (\string"ALPHA\string")
7474     :min-range-length \@glsminrange)}}%
7475 \write\glswrite{(define-location-class
7476     \string"Appendix-page-numbers\string"^^J\space\space\space
7477     (\string"ALPHA\string"
7478     :sep \string"\@glsAlphacompositor\string"
7479     \string"arabic-numbers\string")
7480     :min-range-length \@glsminrange)}}%
7481 \write\glswrite{(define-location-class
7482     \string"arabic-section-numbers\string"^^J\space\space\space

```



```

7483      (\string"arabic-numbers\string"
7484       :sep \string"\glscompositor\string"
7485       \string"arabic-numbers\string")
7486       :min-range-length \@glsminrange))}%
7487 \write\glswrite{^^J; user defined location classes}%
7488 \write\glswrite{\@xdyuserlocationdefs}%
7489 \write\glswrite{^^J; define cross-reference class^^J}%
7490 \write\glswrite{(define-crossref-class \string"see\string"
7491   :unverified )}%
7492 \write\glswrite{(markup-crossref-list
7493   :class \string"see\string"^^J\space\space\space
7494   :open \string"\string\glsseeformat\string"
7495   :close \string"{}\string"))}%
7496 \write\glswrite{^^J; define the order of the location classes}%
7497 \write\glswrite{(define-location-class-order
7498   (\@xdylocationclassorder))}%
7499 \write\glswrite{^^J; define the glossary markup^^J}%
7500 \write\glswrite{(markup-index^^J\space\space\space
7501   :open \string"\string
7502     \glossarysection[\string\glossarytoctitle]{\string
7503     \glossarytitle}\string\glossarypreamble\string~\n\string\begin
7504     {theglossary}\string\glossaryheader\string~\n\string" ^^J\space
7505     \space\space:close \string"\expandafter\@gobble
7506     \string\%\string~\n\string
7507     \end{theglossary}\string\glossarypostamble
7508     \string~\n\string" ^^J\space\space\space
7509     :tree)}}%
7510 \write\glswrite{(markup-letter-group-list
7511   :sep \string"\string\glsgroupskip\string~\n\string"))}%
7512 \write\glswrite{(markup-indexentry
7513   :open \string"\string\relax \string\glsresetentrylist
7514     \string~\n\string"))}%
7515 \write\glswrite{(markup-locclass-list :open
7516   \string"\glsopenbrace\string\glossaryentrynumbers
7517   \glsopenbrace\string\relax\space \string"^^J\space\space\space
7518   :sep \string", \string"
7519   :close \string"\glsclosebrace\glsclosebrace\string"))}%
7520 \write\glswrite{(markup-locref-list
7521   :sep \string"\string\delimN\space\string"))}%
7522 \write\glswrite{(markup-range
7523   :sep \string"\string\delimR\space\string"))}%
7524 \@onelevel@sanitize\gls@suffiF
7525 \@onelevel@sanitize\gls@suffiFF
7526 \ifx\gls@suffiF\@empty
7527 \else
7528   \write\glswrite{(markup-range
7529     :close "\gls@suffiF" :length 1 :ignore-end)}}%
7530 \fi
7531 \ifx\gls@suffiFF\@empty

```

```

7532 \else
7533 \write\glswrite{(markup-range
7534 :close "\gls@suffixFF" :length 2 :ignore-end)}}%
7535 \fi
7536 \write\glswrite{^^J; define format to use for locations^^J}%
7537 \write\glswrite{\@xdylocref}%
7538 \write\glswrite{^^J; define letter group list format^^J}%
7539 \write\glswrite{(markup-letter-group-list
7540 :sep \string\string\glsgroupskip\string~n\string)}}%
7541 \write\glswrite{^^J; letter group headings^^J}%
7542 \write\glswrite{(markup-letter-group
7543 :open-head \string\string\glsgroupheading
7544 \glsopenbrace\string^^J\space\space\space
7545 :close-head \string\glsclosebrace\string)}}%
7546 \write\glswrite{^^J; additional letter groups^^J}%
7547 \write\glswrite{\@xdylettergroups}%
7548 \write\glswrite{^^J; additional sort rules^^J}
7549 \write\glswrite{\@xdysortrules}%
7550 \noist}
7551 \else
7552 \edef\@gls@actualchar{\string?}
7553 \edef\@gls@encapchar{\string|}
7554 \edef\@gls@levelchar{\string!}
7555 \edef\@gls@quotechar{\string"}
7556 \def\writeist{\relax
7557 \openout\glswrite=\istfilename
7558 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
7559 created by the glossaries package}
7560 \write\glswrite{\expandafter\@gobble\string\% for document
7561 '\jobname' on \the\year-\the\month-\the\day}
7562 \write\glswrite{actual '\@gls@actualchar'}
7563 \write\glswrite{encap '\@gls@encapchar'}
7564 \write\glswrite{level '\@gls@levelchar'}
7565 \write\glswrite{quote '\@gls@quotechar'}
7566 \write\glswrite{keyword \string\string\glossaryentry\string"}
7567 \write\glswrite{preamble \string\string\glossarysection[\string
7568 \glossarytoctitle]{\string\glossarytitle}\string
7569 \glossarypreamble\string\n\string\begin{theglossary}\string
7570 \glossaryheader\string\n\string"}
7571 \write\glswrite{postamble \string\string\%\string\n\string
7572 \end{theglossary}\string\glossarypostamble\string\n
7573 \string"}
7574 \write\glswrite{group_skip \string\string\glsgroupskip\string\n
7575 \string"}
7576 \write\glswrite{item_0 \string\string\%\string\n\string"}
7577 \write\glswrite{item_1 \string\string\%\string\n\string"}
7578 \write\glswrite{item_2 \string\string\%\string\n\string"}
7579 \write\glswrite{item_01 \string\string\%\string\n\string"}
7580 \write\glswrite{item_x1

```

```

7581     \string"\string\relax \string\glsresetentrylist\string\n
7582     \string"}
7583     \write\glswrite{item_12 \string"\string%\string\n\string"}
7584     \write\glswrite{item_x2
7585     \string"\string\relax \string\glsresetentrylist\string\n
7586     \string"}
7587     \write\glswrite{delim_0 \string"\string\{\string
7588     \glossaryentrynumbers\string\{\string\relax \string"}
7589     \write\glswrite{delim_1 \string"\string\{\string
7590     \glossaryentrynumbers\string\{\string\relax \string"}
7591     \write\glswrite{delim_2 \string"\string\{\string
7592     \glossaryentrynumbers\string\{\string\relax \string"}
7593     \write\glswrite{delim_t \string"\string\}\string\}\string"}
7594     \write\glswrite{delim_n \string"\string\delimN \string"}
7595     \write\glswrite{delim_r \string"\string\delimR \string"}
7596     \write\glswrite{headings_flag 1}
7597     \write\glswrite{heading_prefix
7598     \string"\string\glsgruppeheading\string\{\string"}
7599     \write\glswrite{heading_suffix
7600     \string"\string\}\string\relax
7601     \string\glsresetentrylist \string"}
7602     \write\glswrite{symhead_positive \string"glssymbols\string"}
7603     \write\glswrite{numhead_positive \string"glssymbols\string"}
7604     \write\glswrite{page_compositor \string"glscpositor\string"}
7605     \@gls@escbsdq\gls@suffixF
7606     \@gls@escbsdq\gls@suffixFF
7607     \ifx\gls@suffixF\@empty
7608     \else
7609     \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
7610     \fi
7611     \ifx\gls@suffixFF\@empty
7612     \else
7613     \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
7614     \fi
7615     \noist
7616   }
7617 \fi

```

\noist

```
7618 \renewcommand*{\noist}{\let\writeist\relax}
```

Compatibility macros.

```
7619 \NeedsTeXFormat{LaTeX2e}
```

```
7620 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]
```

Compatibility macros for predefined glossary styles:

`compatglossarystyle` Defines a compatibility glossary style.

```
7621 \newcommand{\compatglossarystyle}[2]{%
```

```
7622   \ifcsundef{@glscompstyle@#1}%
```

```

7623  {%
7624    \csdef{@glscompstyle@#1}{#2}%
7625  }%
7626  {%
7627    \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
7628  }%
7629 }

```

Backward compatible inline style.

```

7630 \compatglossarystyle{inline}{%
7631   \renewcommand{\glossaryentryfield}[5]{%
7632     \glsinlinedopostchild
7633     \gls@inlinesep
7634     \def\glo@desc{##3}%
7635     \def\@no@post@desc{\nopostdesc}%
7636     \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
7637     \ifx\glo@desc\@no@post@desc
7638       \glsinlineemptydescformat{##4}{##5}%
7639     \else
7640       \ifstrempy{##3}%
7641         {\glsinlineemptydescformat{##4}{##5}}%
7642         {\glsinlinedescformat{##3}{##4}{##5}}%
7643     \fi
7644     \ifglshaschildren{##1}%
7645     {%
7646       \glsresetsubentrycounter
7647       \glsinlineparentchildseparator
7648       \def\gls@inlinesubsep{}%
7649       \def\gls@inlinepostchild{\glsinlinepostchild}%
7650     }%
7651   }%
7652   \def\gls@inlinesep{\glsinlineseparator}%
7653 }%

```

Sub-entries display description:

```

7654 \renewcommand{\glossarysubentryfield}[6]{%
7655   \gls@inlinesubsep%
7656   \glsinlinesubnameformat{##2}{##3}%
7657   \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
7658   \def\gls@inlinesubsep{\glsinlinesubseparator}%
7659 }%
7660 }

```

Backward compatible list style.

```

7661 \compatglossarystyle{list}{%
7662   \renewcommand*{\glossaryentryfield}[5]{%
7663     \item[\glsentryitem{##1}\glstarget{##1}{##2}]
7664     ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

7665 \renewcommand*{\glossarysubentryfield}[6]{%

```

```

7666 \glssubentryitem{##2}%
7667 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
7668 }

```

Backward compatible listgroup style.

```

7669 \compatglossarystyle{listgroup}{%
7670 \csuse{@glscmpstyle@list}%
7671 }%

```

Backward compatible listhypergroup style.

```

7672 \compatglossarystyle{listhypergroup}{%
7673 \csuse{@glscmpstyle@list}%
7674 }%

```

Backward compatible altlist style.

```

7675 \compatglossarystyle{altlist}{%
7676 \renewcommand*{\glossaryentryfield}[5]{%
7677 \item[\glssentryitem{##1}\glstarget{##1}{##2}]%
7678 \mbox{}\par\nobreak\@afterheading
7679 ##3\glspostdescription\space ##5}%
7680 \renewcommand{\glossarysubentryfield}[6]{%
7681 \par
7682 \glssubentryitem{##2}%
7683 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
7684 }%

```

Backward compatible altlistgroup style.

```

7685 \compatglossarystyle{altlistgroup}{%
7686 \csuse{@glscmpstyle@altlist}%
7687 }%

```

Backward compatible altlisthypergroup style.

```

7688 \compatglossarystyle{altlisthypergroup}{%
7689 \csuse{@glscmpstyle@altlist}%
7690 }%

```

Backward compatible listdotted style.

```

7691 \compatglossarystyle{listdotted}{%
7692 \renewcommand*{\glossaryentryfield}[5]{%
7693 \item[\makebox[\glslistdottedwidth][l]{%
7694 \glssentryitem{##1}\glstarget{##1}{##2}%
7695 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
7696 \renewcommand*{\glossarysubentryfield}[6]{%
7697 \item[\makebox[\glslistdottedwidth][l]{%
7698 \glssubentryitem{##2}%
7699 \glstarget{##2}{##3}%
7700 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
7701 }%

```

Backward compatible sublistdotted style.

```

7702 \compatglossarystyle{sublistdotted}{%
7703 \csuse{@glscmpstyle@listdotted}%

```

```

7704 \renewcommand*{\glossaryentryfield}[5]{%
7705 \item[\glentryitem{##1}\glstarget{##1}{##2}]}%
7706 }%

```

Backward compatible long style.

```

7707 \compatglossarystyle{long}{%
7708 \renewcommand*{\glossaryentryfield}[5]{%
7709 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
7710 \renewcommand*{\glossarysubentryfield}[6]{%
7711 &
7712 \glssubentryitem{##2}%
7713 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
7714 }%

```

Backward compatible longborder style.

```

7715 \compatglossarystyle{longborder}{%
7716 \csuse{@glscmpstyle@long}%
7717 }%

```

Backward compatible longheader style.

```

7718 \compatglossarystyle{longheader}{%
7719 \csuse{@glscmpstyle@long}%
7720 }%

```

Backward compatible longheaderborder style.

```

7721 \compatglossarystyle{longheaderborder}{%
7722 \csuse{@glscmpstyle@long}%
7723 }%

```

Backward compatible long3col style.

```

7724 \compatglossarystyle{long3col}{%
7725 \renewcommand*{\glossaryentryfield}[5]{%
7726 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
7727 \renewcommand*{\glossarysubentryfield}[6]{%
7728 &
7729 \glssubentryitem{##2}%
7730 \glstarget{##2}{\strut}##4 & ##6\\}%
7731 }%

```

Backward compatible long3colborder style.

```

7732 \compatglossarystyle{long3colborder}{%
7733 \csuse{@glscmpstyle@long3col}%
7734 }%

```

Backward compatible long3colheader style.

```

7735 \compatglossarystyle{long3colheader}{%
7736 \csuse{@glscmpstyle@long3col}%
7737 }%

```

Backward compatible long3colheaderborder style.

```

7738 \compatglossarystyle{long3colheaderborder}{%
7739 \csuse{@glscmpstyle@long3col}%
7740 }%

```

Backward compatible long4col style.

```
7741 \compatglossarystyle{long4col}{%
7742   \renewcommand*{\glossaryentryfield}[5]{%
7743     \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
7744   \renewcommand*{\glossarysubentryfield}[6]{%
7745     &
7746     \glssubentryitem{##2}%
7747     \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
7748 }%
```

Backward compatible long4colheader style.

```
7749 \compatglossarystyle{long4colheader}{%
7750   \csuse{@glscmpstyle@long4col}%
7751 }%
```

Backward compatible long4colborder style.

```
7752 \compatglossarystyle{long4colborder}{%
7753   \csuse{@glscmpstyle@long4col}%
7754 }%
```

Backward compatible long4colheaderborder style.

```
7755 \compatglossarystyle{long4colheaderborder}{%
7756   \csuse{@glscmpstyle@long4col}%
7757 }%
```

Backward compatible altlong4col style.

```
7758 \compatglossarystyle{altlong4col}{%
7759   \csuse{@glscmpstyle@long4col}%
7760 }%
```

Backward compatible altlong4colheader style.

```
7761 \compatglossarystyle{altlong4colheader}{%
7762   \csuse{@glscmpstyle@long4col}%
7763 }%
```

Backward compatible altlong4colborder style.

```
7764 \compatglossarystyle{altlong4colborder}{%
7765   \csuse{@glscmpstyle@long4col}%
7766 }%
```

Backward compatible altlong4colheaderborder style.

```
7767 \compatglossarystyle{altlong4colheaderborder}{%
7768   \csuse{@glscmpstyle@long4col}%
7769 }%
```

Backward compatible long style.

```
7770 \compatglossarystyle{longragged}{%
7771   \renewcommand*{\glossaryentryfield}[5]{%
7772     \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
7773     \tabularnewline}%
7774   \renewcommand*{\glossarysubentryfield}[6]{%
7775     &
```

```

7776 \glssubentryitem{##2}%
7777 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
7778 \tabularnewline}%
7779 }%

```

Backward compatible longraggedborder style.

```

7780 \compatglossarystyle{longraggedborder}{%
7781 \csuse{@glscmpstyle@longragged}%
7782 }%

```

Backward compatible longraggedheader style.

```

7783 \compatglossarystyle{longraggedheader}{%
7784 \csuse{@glscmpstyle@longragged}%
7785 }%

```

Backward compatible longraggedheaderborder style.

```

7786 \compatglossarystyle{longraggedheaderborder}{%
7787 \csuse{@glscmpstyle@longragged}%
7788 }%

```

Backward compatible longragged3col style.

```

7789 \compatglossarystyle{longragged3col}{%
7790 \renewcommand*{\glossaryentryfield}[5]{%
7791 \glssubentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
7792 \renewcommand*{\glossarysubentryfield}[6]{%
7793 &
7794 \glssubentryitem{##2}%
7795 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
7796 }%

```

Backward compatible longragged3colborder style.

```

7797 \compatglossarystyle{longragged3colborder}{%
7798 \csuse{@glscmpstyle@longragged3col}%
7799 }%

```

Backward compatible longragged3colheader style.

```

7800 \compatglossarystyle{longragged3colheader}{%
7801 \csuse{@glscmpstyle@longragged3col}%
7802 }%

```

Backward compatible longragged3colheaderborder style.

```

7803 \compatglossarystyle{longragged3colheaderborder}{%
7804 \csuse{@glscmpstyle@longragged3col}%
7805 }%

```

Backward compatible altlongragged4col style.

```

7806 \compatglossarystyle{altlongragged4col}{%
7807 \renewcommand*{\glossaryentryfield}[5]{%
7808 \glssubentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
7809 \renewcommand*{\glossarysubentryfield}[6]{%
7810 &
7811 \glssubentryitem{##2}%

```



```

7812 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
7813 }%

```

Backward compatible altlongragged4colheader style.

```

7814 \compatglossarystyle{altlongragged4colheader}{%
7815 \csuse{@glscmpstyle@altlong4col}%
7816 }%

```

Backward compatible altlongragged4colborder style.

```

7817 \compatglossarystyle{altlongragged4colborder}{%
7818 \csuse{@glscmpstyle@altlong4col}%
7819 }%

```

Backward compatible altlongragged4colheaderborder style.

```

7820 \compatglossarystyle{altlongragged4colheaderborder}{%
7821 \csuse{@glscmpstyle@altlong4col}%
7822 }%

```

Backward compatible index style.

```

7823 \compatglossarystyle{index}{%
7824 \renewcommand*{\glossaryentryfield}[5]{%
7825 \item\glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
7826 \ifx\relax##4\relax
7827 \else
7828 \space(##4)%
7829 \fi
7830 \space ##3\glspostdescription \space ##5}%
7831 \renewcommand*{\glossarysubentryfield}[6]{%
7832 \ifcase##1\relax
7833 % level 0
7834 \item
7835 \or
7836 % level 1
7837 \subitem
7838 \glssubentryitem{##2}%
7839 \else
7840 % all other levels
7841 \subsubitem
7842 \fi
7843 \textbf{\glstarget{##2}{##3}}%
7844 \ifx\relax##5\relax
7845 \else
7846 \space(##5)%
7847 \fi
7848 \space##4\glspostdescription\space ##6}%
7849 }%

```

Backward compatible indexgroup style.

```

7850 \compatglossarystyle{indexgroup}{%
7851 \csuse{@glscmpstyle@index}%
7852 }%

```

Backward compatible indexhypergroup style.

```
7853 \compatglossarystyle{indexhypergroup}{%
7854 \csuse{@glscmpstyle@index}%
7855 }%
```

Backward compatible tree style.

```
7856 \compatglossarystyle{tree}{%
7857 \renewcommand{\glossaryentryfield}[5]{%
7858 \hangindent0pt\relax
7859 \parindent0pt\relax
7860 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
7861 \ifx\relax##4\relax
7862 \else
7863 \space{##4}%
7864 \fi
7865 \space ##3\glspostdescription \space ##5\par}%
7866 \renewcommand{\glossarysubentryfield}[6]{%
7867 \hangindent##1\glstreeindent\relax
7868 \parindent##1\glstreeindent\relax
7869 \ifnum##1=1\relax
7870 \glssubentryitem{##2}%
7871 \fi
7872 \textbf{\glstarget{##2}{##3}}%
7873 \ifx\relax##5\relax
7874 \else
7875 \space{##5}%
7876 \fi
7877 \space##4\glspostdescription\space ##6\par}%
7878 }%
```

Backward compatible treegroup style.

```
7879 \compatglossarystyle{treegroup}{%
7880 \csuse{@glscmpstyle@tree}%
7881 }%
```

Backward compatible treehypergroup style.

```
7882 \compatglossarystyle{treehypergroup}{%
7883 \csuse{@glscmpstyle@tree}%
7884 }%
```

Backward compatible treenoname style.

```
7885 \compatglossarystyle{treenoname}{%
7886 \renewcommand{\glossaryentryfield}[5]{%
7887 \hangindent0pt\relax
7888 \parindent0pt\relax
7889 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
7890 \ifx\relax##4\relax
7891 \else
7892 \space{##4}%
7893 \fi
7894 \space ##3\glspostdescription \space ##5\par}%

```

```

7895 \renewcommand{\glossarysubentryfield}[6]{%
7896   \hangindent##1\glstreeindent\relax
7897   \parindent##1\glstreeindent\relax
7898   \ifnum##1=1\relax
7899     \glssubentryitem{##2}%
7900   \fi
7901   \glstarget{##2}{\strut}%
7902   ##4\glspostdescription\space ##6\par}%
7903 }%

```

Backward compatible treenonamegroup style.

```

7904 \compatglossarystyle{treenonamegroup}{%
7905   \csuse{@glscmpstyle@treenoname}%
7906 }%

```

Backward compatible treenonamehypergroup style.

```

7907 \compatglossarystyle{treenonamehypergroup}{%
7908   \csuse{@glscmpstyle@treenoname}%
7909 }%

```

Backward compatible alttree style.

```

7910 \compatglossarystyle{alttree}{%
7911   \renewcommand{\glossaryentryfield}[5]{%
7912     \ifnum \@gls@prevlevel=0\relax
7913     \else
7914       \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
7915       \hangindent\glstreeindent
7916       \parindent\glstreeindent
7917     \fi
7918     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
7919       \glssubentryitem{##1}\textbf{\glstarget{##1}{##2}}}%
7920     \ifx\relax##4\relax
7921     \else
7922       (##4)\space
7923     \fi
7924     ##3\glspostdescription\space ##5\par
7925     \def\@gls@prevlevel{0}%
7926   }%
7927   \renewcommand{\glossarysubentryfield}[6]{%
7928     \ifnum##1=1\relax
7929       \glssubentryitem{##2}%
7930     \fi
7931     \ifnum \@gls@prevlevel=##1\relax
7932     \else
7933       \@ifundefined{@glswidestname\romannumeral##1}{%
7934         \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
7935         \settowidth{\gls@tmplen}{\textbf{%
7936           \csname @glswidestname\romannumeral##1\endcsname\space}}}%
7937       \ifnum \@gls@prevlevel<##1\relax
7938         \setlength\glstreeindent\gls@tmplen
7939         \addtolength\glstreeindent\parindent

```

```

7940         \parindent\glstreeindent
7941     \else
7942         \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
7943             \settowidth{\glstreeindent}{\textbf{%
7944                 \@glswidestname\space}}}{%
7945             \settowidth{\glstreeindent}{\textbf{%
7946                 \csname @glswidestname\romannumeral\@gls@prevlevel
7947                     \endcsname\space}}}{%
7948             \addtolength\parindent{-\glstreeindent}%
7949             \setlength\glstreeindent\parindent
7950         \fi
7951     \fi
7952     \hangindent\glstreeindent
7953     \makebox[0pt][r]{\makebox[\@gls@tmplen][l]{%
7954         \textbf{\glstarget{##2}{##3}}}%
7955     \ifx##5\relax\relax
7956     \else
7957         (##5)\space
7958     \fi
7959     ##4\glspostdescription\space ##6\par
7960     \def\@gls@prevlevel{##1}%
7961 }%
7962 }%

```

Backward compatible alttreegroup style.

```

7963 \compatglossarystyle{alttreegroup}{%
7964 \csuse{@glscompstyle@alttree}%
7965 }%

```

Backward compatible alttreehypergroup style.

```

7966 \compatglossarystyle{alttreehypergroup}{%
7967 \csuse{@glscompstyle@alttree}%
7968 }%

```

Backward compatible mcolindex style.

```

7969 \compatglossarystyle{mcolindex}{%
7970 \csuse{@glscompstyle@index}%
7971 }%

```

Backward compatible mcolindexgroup style.

```

7972 \compatglossarystyle{mcolindexgroup}{%
7973 \csuse{@glscompstyle@index}%
7974 }%

```

Backward compatible mcolindexhypergroup style.

```

7975 \compatglossarystyle{mcolindexhypergroup}{%
7976 \csuse{@glscompstyle@index}%
7977 }%

```

Backward compatible mcoltree style.

```

7978 \compatglossarystyle{mcoltree}{%
7979 \csuse{@glscompstyle@tree}%

```

7980 }%

Backward compatible mcoltreegroup style.

7981 \compatglossarystyle{mcolindextreegroup}{%

7982 \csuse{@glscmpstyle@tree}%

7983 }%

Backward compatible mcoltreehypergroup style.

7984 \compatglossarystyle{mcolindextreehypergroup}{%

7985 \csuse{@glscmpstyle@tree}%

7986 }%

Backward compatible mcoltreenoname style.

7987 \compatglossarystyle{mcoltreenoname}{%

7988 \csuse{@glscmpstyle@tree}%

7989 }%

Backward compatible mcoltreenonamegroup style.

7990 \compatglossarystyle{mcoltreenonamegroup}{%

7991 \csuse{@glscmpstyle@tree}%

7992 }%

Backward compatible mcoltreenonamehypergroup style.

7993 \compatglossarystyle{mcoltreenonamehypergroup}{%

7994 \csuse{@glscmpstyle@tree}%

7995 }%

Backward compatible mcolalmtree style.

7996 \compatglossarystyle{mcolalmtree}{%

7997 \csuse{@glscmpstyle@almtree}%

7998 }%

Backward compatible mcolalmtreegroup style.

7999 \compatglossarystyle{mcolalmtreegroup}{%

8000 \csuse{@glscmpstyle@almtree}%

8001 }%

Backward compatible mcolalmtreehypergroup style.

8002 \compatglossarystyle{mcolalmtreehypergroup}{%

8003 \csuse{@glscmpstyle@almtree}%

8004 }%

Backward compatible superragged style.

8005 \compatglossarystyle{superragged}{%

8006 \renewcommand*{\glossaryentryfield}[5]{%

8007 \glstarget{##1}{##2} & ##3\glspostdescription\space ##5%

8008 \tabularnewline}%

8009 \renewcommand*{\glossarysubentryfield}[6]{%

8010 &

8011 \glssubentryitem{##2}%

8012 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%

8013 \tabularnewline}%

8014 }%

Backward compatible superraggedborder style.

```
8015 \compatglossarystyle{superraggedborder}{%  
8016 \csuse{@glscmpstyle@superragged}%  
8017 }%
```

Backward compatible superraggedheader style.

```
8018 \compatglossarystyle{superraggedheader}{%  
8019 \csuse{@glscmpstyle@superragged}%  
8020 }%
```

Backward compatible superraggedheaderborder style.

```
8021 \compatglossarystyle{superraggedheaderborder}{%  
8022 \csuse{@glscmpstyle@superragged}%  
8023 }%
```

Backward compatible superragged3col style.

```
8024 \compatglossarystyle{superragged3col}{%  
8025 \renewcommand*{\glossaryentryfield}[5]{%  
8026 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%  
8027 \renewcommand*{\glossarysubentryfield}[6]{%  
8028 &  
8029 \glssubentryitem{##2}%  
8030 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%  
8031 }%
```

Backward compatible superragged3colborder style.

```
8032 \compatglossarystyle{superragged3colborder}{%  
8033 \csuse{@glscmpstyle@superragged3col}%  
8034 }%
```

Backward compatible superragged3colheader style.

```
8035 \compatglossarystyle{superragged3colheader}{%  
8036 \csuse{@glscmpstyle@superragged3col}%  
8037 }%
```

Backward compatible superragged3colheaderborder style.

```
8038 \compatglossarystyle{superragged3colheaderborder}{%  
8039 \csuse{@glscmpstyle@superragged3col}%  
8040 }%
```

Backward compatible altsuperragged4col style.

```
8041 \compatglossarystyle{altsuperragged4col}{%  
8042 \renewcommand*{\glossaryentryfield}[5]{%  
8043 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%  
8044 \renewcommand*{\glossarysubentryfield}[6]{%  
8045 &  
8046 \glssubentryitem{##2}%  
8047 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%  
8048 }%
```

Backward compatible altsuperragged4colheader style.

```
8049 \compatglossarystyle{altsuperragged4colheader}{%
```

```
8050 \csuse{@glscompstyle@altsuperragged4col}%
8051 }%
```

Backward compatible altsuperragged4colborder style.

```
8052 \compatglossarystyle{altsuperragged4colborder}{%
8053 \csuse{@glscompstyle@altsuperragged4col}%
8054 }%
```

Backward compatible altsuperragged4colheaderborder style.

```
8055 \compatglossarystyle{altsuperragged4colheaderborder}{%
8056 \csuse{@glscompstyle@altsuperragged4col}%
8057 }%
```

Backward compatible super style.

```
8058 \compatglossarystyle{super}{%
8059 \renewcommand*{\glossaryentryfield}[5]{%
8060 \glstarget{##1}{\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8061 \renewcommand*{\glossarysubentryfield}[6]{%
8062 &
8063 \glssubentryitem{##2}%
8064 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8065 }%
```

Backward compatible superborder style.

```
8066 \compatglossarystyle{superborder}{%
8067 \csuse{@glscompstyle@super}%
8068 }%
```

Backward compatible superheader style.

```
8069 \compatglossarystyle{superheader}{%
8070 \csuse{@glscompstyle@super}%
8071 }%
```

Backward compatible superheaderborder style.

```
8072 \compatglossarystyle{superheaderborder}{%
8073 \csuse{@glscompstyle@super}%
8074 }%
```

Backward compatible super3col style.

```
8075 \compatglossarystyle{super3col}{%
8076 \renewcommand*{\glossaryentryfield}[5]{%
8077 \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\\}%
8078 \renewcommand*{\glossarysubentryfield}[6]{%
8079 &
8080 \glssubentryitem{##2}%
8081 \glstarget{##2}{\strut}##4 & ##6\\}%
8082 }%
```

Backward compatible super3colborder style.

```
8083 \compatglossarystyle{super3colborder}{%
8084 \csuse{@glscompstyle@super3col}%
8085 }%
```

Backward compatible super3colheader style.

```
8086 \compatglossarystyle{super3colheader}{%  
8087 \csuse{@glscompstyle@super3col}%  
8088 }%
```

Backward compatible super3colheaderborder style.

```
8089 \compatglossarystyle{super3colheaderborder}{%  
8090 \csuse{@glscompstyle@super3col}%  
8091 }%
```

Backward compatible super4col style.

```
8092 \compatglossarystyle{super4col}{%  
8093 \renewcommand*{\glossaryentryfield}[5]{%  
8094 \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%  
8095 \renewcommand*{\glossarysubentryfield}[6]{%  
8096 &  
8097 \glssubentryitem{##2}%  
8098 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%  
8099 }%
```

Backward compatible super4colheader style.

```
8100 \compatglossarystyle{super4colheader}{%  
8101 \csuse{@glscompstyle@super4col}%  
8102 }%
```

Backward compatible super4colborder style.

```
8103 \compatglossarystyle{super4colborder}{%  
8104 \csuse{@glscompstyle@super4col}%  
8105 }%
```

Backward compatible super4colheaderborder style.

```
8106 \compatglossarystyle{super4colheaderborder}{%  
8107 \csuse{@glscompstyle@super4col}%  
8108 }%
```

Backward compatible altsuper4col style.

```
8109 \compatglossarystyle{altsuper4col}{%  
8110 \csuse{@glscompstyle@super4col}%  
8111 }%
```

Backward compatible altsuper4colheader style.

```
8112 \compatglossarystyle{altsuper4colheader}{%  
8113 \csuse{@glscompstyle@super4col}%  
8114 }%
```

Backward compatible altsuper4colborder style.

```
8115 \compatglossarystyle{altsuper4colborder}{%  
8116 \csuse{@glscompstyle@super4col}%  
8117 }%
```

Backward compatible altsuper4colheaderborder style.

```
8118 \compatglossarystyle{altsuper4colheaderborder}{%  
8119 \csuse{@glscompstyle@super4col}%  
8120 }%
```


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.

```
8121 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
8122 \ProvidesPackage{glossaries-accsupp}[2013/11/14 v4.0 (NLCT)]
```

```
8123 Experimental glossaries accessibility]
```

Pass all options to glossaries:

```
8124 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
8125 \ProcessOptions
```

Override style compatibility macros:

```
8126 \newcommand*{\compatibleglossentry}[2]{%
```

```
8127   \toks@{#2}%
```

```
8128   \protected@edef\@do@glossentry{%
```

```
8129     \noexpand\accsuppglossaryentryfield{#1}%
```

```
8130     {\noexpand\glsnamefont
```

```
8131       {\expandafter\expandonce\csname glo@#1@name\endcsname}}%
```

```
8132     {\expandafter\expandonce\csname glo@#1@desc\endcsname}}%
```

```
8133     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}}%
```

```
8134     {\the\toks@}%
```

```
8135   }%
```

```
8136   \@do@glossentry
```

```
8137 }
```

```
8138 \newcommand*{\compatiblesubglossentry}[3]{%
```

```
8139   \toks@{#3}%
```

```
8140   \protected@edef\@do@subglossentry{%
```

```
8141     \noexpand\accsuppglossarysubentryfield{\number#1}%
```

```
8142     {#2}%
```

```
8143     {\noexpand\glsnamefont
```

```
8144       {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
```

```
8145     {\expandafter\expandonce\csname glo@#2@desc\endcsname}}%
```

```
8146     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}}%
```

```
8147     {\the\toks@}%
```

```
8148   }%
```

```
8149   \@do@subglossentry
```

```
8150 }
```

Required packages:

```
8151 \RequirePackage{glossaries}
```

```
8152 \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:

```
8153 \define@key{glossentry}{access}{%  
8154   \def\@glo@access{#1}%  
8155 }
```

textaccess The replacement text corresponding to the text key:

```
8156 \define@key{glossentry}{textaccess}{%  
8157   \def\@glo@textaccess{#1}%  
8158 }
```

firstaccess The replacement text corresponding to the first key:

```
8159 \define@key{glossentry}{firstaccess}{%  
8160   \def\@glo@firstaccess{#1}%  
8161 }
```

pluralaccess The replacement text corresponding to the plural key:

```
8162 \define@key{glossentry}{pluralaccess}{%  
8163   \def\@glo@pluralaccess{#1}%  
8164 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
8165 \define@key{glossentry}{firstpluralaccess}{%  
8166   \def\@glo@firstpluralaccess{#1}%  
8167 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
8168 \define@key{glossentry}{symbolaccess}{%  
8169   \def\@glo@symbolaccess{#1}%  
8170 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
8171 \define@key{glossentry}{symbolpluralaccess}{%  
8172   \def\@glo@symbolpluralaccess{#1}%  
8173 }
```

descriptionaccess The replacement text corresponding to the description key:

```
8174 \define@key{glossentry}{descriptionaccess}{%  
8175   \def\@glo@descaccess{#1}%  
8176 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
8177 \define@key{glossentry}{descriptionpluralaccess}{%
8178   \def\@glo@descpluralaccess{#1}%
8179 }
```

shortaccess The replacement text corresponding to the short key:

```
8180 \define@key{glossentry}{shortaccess}{%
8181   \def\@glo@shortaccess{#1}%
8182 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
8183 \define@key{glossentry}{shortpluralaccess}{%
8184   \def\@glo@shortpluralaccess{#1}%
8185 }
```

longaccess The replacement text corresponding to the long key:

```
8186 \define@key{glossentry}{longaccess}{%
8187   \def\@glo@longaccess{#1}%
8188 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
8189 \define@key{glossentry}{longpluralaccess}{%
8190   \def\@glo@longpluralaccess{#1}%
8191 }
```

There are no equivalent keys for the user1...user6 keys. The replacement text would have to be explicitly put in the value, e.g., user1={\glsaccsupp{inches}{in}}.

Append these new keys to \@gls@keymap:

```
8192 \appto\@gls@keymap{,%
8193   {access}{access},%
8194   {textaccess}{textaccess},%
8195   {firstaccess}{firstaccess},%
8196   {pluralaccess}{pluralaccess},%
8197   {firstpluralaccess}{firstpluralaccess},%
8198   {symbolaccess}{symbolaccess},%
8199   {symbolpluralaccess}{symbolpluralaccess},%
8200   {descaccess}{descaccess},%
8201   {descpluralaccess}{descpluralaccess},%
8202   {shortaccess}{shortaccess},%
8203   {shortpluralaccess}{shortpluralaccess},%
8204   {longaccess}{longaccess},%
8205   {longpluralaccess}{longpluralaccess}%
8206 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
8207 \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):

```
8208 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
8209 \renewcommand*{\@newglossaryentryprehook}{%
8210   \@gls@oldnewglossaryentryprehook
8211   \def\@glo@access{\@glo@symbol}%
```

Initialise the other keys:

```
8212   \def\@glo@textaccess{\@glo@access}%
8213   \def\@glo@firstaccess{\@glo@access}%
8214   \def\@glo@pluralaccess{\@glo@textaccess}%
8215   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
8216   \def\@glo@symbolaccess{\relax}%
8217   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
8218   \def\@glo@descaccess{\relax}%
8219   \def\@glo@descpluralaccess{\@glo@descaccess}%
8220   \def\@glo@shortaccess{\relax}%
8221   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
8222   \def\@glo@longaccess{\relax}%
8223   \def\@glo@longpluralaccess{\@glo@longaccess}%
8224 }
```

Add to the end hook:

```
8225 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
8226 \renewcommand*{\@newglossaryentryposthook}{%
8227   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
8228   \expandafter
8229     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
8230       \@glo@access}%
8231   \expandafter
8232     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
8233       \@glo@textaccess}%
8234   \expandafter
8235     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
8236       \@glo@firstaccess}%
8237   \expandafter
8238     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
8239       \@glo@pluralaccess}%
8240   \expandafter
8241     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
8242       \@glo@firstpluralaccess}%
8243   \expandafter
8244     \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
8245       \@glo@symbolaccess}%
8246   \expandafter
8247     \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
8248       \@glo@symbolpluralaccess}%
8249   \expandafter
```

```

8250 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
8251 \@glo@descaccess}%
8252 \expandafter
8253 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
8254 \@glo@descpluralaccess}%
8255 \expandafter
8256 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
8257 \@glo@shortaccess}%
8258 \expandafter
8259 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
8260 \@glo@shortpluralaccess}%
8261 \expandafter
8262 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
8263 \@glo@longaccess}%
8264 \expandafter
8265 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
8266 \@glo@longpluralaccess}%
8267 }

```

6.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

8268 \newcommand*{\glsentryaccess}[1]{%
8269 \csname glo@#1@access\endcsname
8270 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

8271 \newcommand*{\glsentrytextaccess}[1]{%
8272 \csname glo@#1@textaccess\endcsname
8273 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

8274 \newcommand*{\glsentryfirstaccess}[1]{%
8275 \csname glo@#1@firstaccess\endcsname
8276 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

8277 \newcommand*{\glsentrypluralaccess}[1]{%
8278 \csname glo@#1@pluralaccess\endcsname
8279 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

8280 \newcommand*{\glsentryfirstpluralaccess}[1]{%
8281 \csname glo@#1@firstpluralaccess\endcsname
8282 }

```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```

8283 \newcommand*{\glsentrysymbolaccess}[1]{%

```

```

8284 \csname glo@#1@symbolpluralaccess\endcsname
8285 }

symbolpluralaccess Get the value of the symbolpluralaccess key for the entry with the given label:
8286 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
8287 \csname glo@#1@symbolpluralaccess\endcsname
8288 }

\glsentrydescaccess Get the value of the descriptionaccess key for the entry with the given label:
8289 \newcommand*{\glsentrydescaccess}[1]{%
8290 \csname glo@#1@descaccess\endcsname
8291 }

trydescpluralaccess Get the value of the descriptionpluralaccess key for the entry with the given label:
8292 \newcommand*{\glsentrydescpluralaccess}[1]{%
8293 \csname glo@#1@descaccess\endcsname
8294 }

glsentryshortaccess Get the value of the shortaccess key for the entry with the given label:
8295 \newcommand*{\glsentryshortaccess}[1]{%
8296 \csname glo@#1@shortaccess\endcsname
8297 }

ryshortpluralaccess Get the value of the shortpluralaccess key for the entry with the given label:
8298 \newcommand*{\glsentryshortpluralaccess}[1]{%
8299 \csname glo@#1@shortpluralaccess\endcsname
8300 }

\glsentrylongaccess Get the value of the longaccess key for the entry with the given label:
8301 \newcommand*{\glsentrylongaccess}[1]{%
8302 \csname glo@#1@longaccess\endcsname
8303 }

trylongpluralaccess Get the value of the longpluralaccess key for the entry with the given label:
8304 \newcommand*{\glsentrylongpluralaccess}[1]{%
8305 \csname glo@#1@longpluralaccess\endcsname
8306 }

\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.)
8307 \newcommand*{\glsaccsupp}[2]{%
8308 \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
8309 }

```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```

8310 \newcommand*{\xglsaccsupp}[2]{%
8311   \protected@edef\@gls@replacementtext{#1}%
8312   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
8313 }

```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```

8314 \DeclareRobustCommand*\glsnameaccessdisplay[2]{%
8315   \protected@edef\@glo@access{\glsentryaccess{#2}}%
8316   \ifx\@glo@access\@gls@noaccess
8317     #1%
8318   \else
8319     \xglsaccsupp{\@glo@access}{#1}%
8320   \fi
8321 }

```

`\glstextaccessdisplay` As above but for the `textaccess` replacement text.

```

8322 \DeclareRobustCommand*\glstextaccessdisplay[2]{%
8323   \protected@edef\@glo@access{\glsentrytextaccess{#2}}%
8324   \ifx\@glo@access\@gls@noaccess
8325     #1%
8326   \else
8327     \xglsaccsupp{\@glo@access}{#1}%
8328   \fi
8329 }

```

`\glspluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```

8330 \DeclareRobustCommand*\glspluralaccessdisplay[2]{%
8331   \protected@edef\@glo@access{\glsentrypluralaccess{#2}}%
8332   \ifx\@glo@access\@gls@noaccess
8333     #1%
8334   \else
8335     \xglsaccsupp{\@glo@access}{#1}%
8336   \fi
8337 }

```

`\glsfirstaccessdisplay` As above but for the `firstaccess` replacement text.

```

8338 \DeclareRobustCommand*\glsfirstaccessdisplay[2]{%
8339   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
8340   \ifx\@glo@access\@gls@noaccess
8341     #1%
8342   \else
8343     \xglsaccsupp{\@glo@access}{#1}%
8344   \fi
8345 }

```

`\glsfirstpluralaccessdisplay` As above but for the `firstpluralaccess` replacement text.

```

8346 \DeclareRobustCommand*{\glsfirstpluralaccessdisplay}[2]{%
8347   \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
8348   \ifx\@glo@access\@gls@noaccess
8349     #1%
8350   \else
8351     \xglsaccsupp{\@glo@access}{#1}%
8352   \fi
8353 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

8354 \DeclareRobustCommand*{\glsymbolaccessdisplay}[2]{%
8355   \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
8356   \ifx\@glo@access\@gls@noaccess
8357     #1%
8358   \else
8359     \xglsaccsupp{\@glo@access}{#1}%
8360   \fi
8361 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

8362 \DeclareRobustCommand*{\glsymbolpluralaccessdisplay}[2]{%
8363   \protected@edef\@glo@access{\glsentrysymbolpluralaccess{#2}}%
8364   \ifx\@glo@access\@gls@noaccess
8365     #1%
8366   \else
8367     \xglsaccsupp{\@glo@access}{#1}%
8368   \fi
8369 }

```

descriptionaccessdisplay As above but for the descriptionaccess replacement text.

```

8370 \DeclareRobustCommand*{\glsdescriptionaccessdisplay}[2]{%
8371   \protected@edef\@glo@access{\glsentrydescaccess{#2}}%
8372   \ifx\@glo@access\@gls@noaccess
8373     #1%
8374   \else
8375     \xglsaccsupp{\@glo@access}{#1}%
8376   \fi
8377 }

```

descriptionpluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

8378 \DeclareRobustCommand*{\glsdescriptionpluralaccessdisplay}[2]{%
8379   \protected@edef\@glo@access{\glsentrydescpluralaccess{#2}}%
8380   \ifx\@glo@access\@gls@noaccess
8381     #1%
8382   \else
8383     \xglsaccsupp{\@glo@access}{#1}%
8384   \fi
8385 }

```


sshortaccessdisplay As above but for the shortaccess replacement text.

```
8386 \DeclareRobustCommand*{\glsshortaccessdisplay}[2]{%
8387   \protected@edef\@glo@access{\glstryshortaccess{#2}}%
8388   \ifx\@glo@access\@gls@noaccess
8389     #1%
8390   \else
8391     \xglsaccsupp{\@glo@access}{#1}%
8392   \fi
8393 }
```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```
8394 \DeclareRobustCommand*{\glsshortpluralaccessdisplay}[2]{%
8395   \protected@edef\@glo@access{\glstryshortpluralaccess{#2}}%
8396   \ifx\@glo@access\@gls@noaccess
8397     #1%
8398   \else
8399     \xglsaccsupp{\@glo@access}{#1}%
8400   \fi
8401 }
```

lslongaccessdisplay As above but for the longaccess replacement text.

```
8402 \DeclareRobustCommand*{\glslongaccessdisplay}[2]{%
8403   \protected@edef\@glo@access{\glstrylongaccess{#2}}%
8404   \ifx\@glo@access\@gls@noaccess
8405     #1%
8406   \else
8407     \xglsaccsupp{\@glo@access}{#1}%
8408   \fi
8409 }
```

pluralaccessdisplay As above but for the longpluralaccess replacement text.

```
8410 \DeclareRobustCommand*{\glslongpluralaccessdisplay}[2]{%
8411   \protected@edef\@glo@access{\glstrylongpluralaccess{#2}}%
8412   \ifx\@glo@access\@gls@noaccess
8413     #1%
8414   \else
8415     \xglsaccsupp{\@glo@access}{#1}%
8416   \fi
8417 }
```

\glsaccessdisplay Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```
8418 \DeclareRobustCommand*{\glsaccessdisplay}[3]{%
8419   \@ifundefined{gls#1accessdisplay}%
8420   {%
8421     \PackageError{glossaries-accsupp}{No accessibility support
8422       for key ‘#1’}{}%
8423   }%
```

```

8424   {%
8425     \csname gls#1accessdisplay\endcsname{#2}{#3}%
8426   }%
8427 }

```

`\gls@default@entryfmt` Redefine the default entry format to use accessibility information

```

8428 \renewcommand*{\@gls@default@entryfmt}[2]{%
8429   \ifdefempty\glscustomtext
8430   {%
8431     \glsifplural
8432     {%
      Plural form
8433       \glscapscase
8434       {%
        Don't adjust case
8435         \ifglsused\glslabel
8436         {%
          Subsequent use
8437           #2{\glspluralaccessdisplay
8438             {\glsentryplural{\glslabel}}{\glslabel}}%
8439           {\glsdescriptionpluralaccessdisplay
8440             {\glsentrydescplural{\glslabel}}{\glslabel}}%
8441           {\glsymbolpluralaccessdisplay
8442             {\glsentrysymbolplural{\glslabel}}{\glslabel}}
8443           {\glsinsert}}%
8444         }%
8445       {%
        First use
8446         #1{\glsfirstpluralaccessdisplay
8447           {\glsentryfirstplural{\glslabel}}{\glslabel}}%
8448           {\glsdescriptionpluralaccessdisplay
8449             {\glsentrydescplural{\glslabel}}{\glslabel}}%
8450           {\glsymbolpluralaccessdisplay
8451             {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
8452           {\glsinsert}}%
8453         }%
8454       }%
8455     {%
      Make first letter upper case
8456       \ifglsused\glslabel
8457       {%
        Subsequent use.
8458         #2{\glspluralaccessdisplay
8459           {\Glsentryplural{\glslabel}}{\glslabel}}%
8460           {\glsdescriptionpluralaccessdisplay
8461             {\Glsentrydescplural{\glslabel}}{\glslabel}}%

```

```

8462         {\glssymbolpluralaccessdisplay
8463         {\glentrysymbolplural{\glslabel}}{\glslabel}}%
8464         {\glsinsert}}%
8465     }%
8466     {%

```

First use

```

8467         #1{\glsfirstpluralaccessdisplay
8468         {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
8469         {\glsdescriptionpluralaccessdisplay
8470         {\glentrydescplural{\glslabel}}{\glslabel}}%
8471         {\glssymbolpluralaccessdisplay
8472         {\glentrysymbolplural{\glslabel}}{\glslabel}}%
8473         {\glsinsert}}%
8474     }%
8475     }%
8476     {%

```

Make all upper case

```

8477         \ifglsused\glslabel
8478         {%

```

Subsequent use

```

8479         \MakeUppercase{%
8480         #2{\glspluralaccessdisplay
8481         {\glentryplural{\glslabel}}{\glslabel}}%
8482         {\glsdescriptionpluralaccessdisplay
8483         {\glentrydescplural{\glslabel}}{\glslabel}}%
8484         {\glssymbolpluralaccessdisplay
8485         {\glentrysymbolplural{\glslabel}}{\glslabel}}%
8486         {\glsinsert}}%
8487     }%
8488     {%

```

First use

```

8489         \MakeUppercase{%
8490         #1{\glsfirstpluralaccessdisplay
8491         {\glentryfirstplural{\glslabel}}{\glslabel}}%
8492         {\glsdescriptionpluralaccessdisplay
8493         {\glentrydescplural{\glslabel}}{\glslabel}}%
8494         {\glssymbolpluralaccessdisplay
8495         {\glentrysymbolplural{\glslabel}}{\glslabel}}%
8496         {\glsinsert}}%
8497     }%
8498     }%
8499     }%
8500     {%

```

Singular form

```

8501         \glscapscase
8502         {%

```

Don't adjust case

```
8503      \ifglsused\glslabel
8504      {%
```

Subsequent use

```
8505      #2{\glstextaccessdisplay
8506          {\glentrytext{\glslabel}}{\glslabel}}%
8507          {\glsdescriptionaccessdisplay
8508              {\glentrydesc{\glslabel}}{\glslabel}}%
8509          {\glssymbolaccessdisplay
8510              {\glentrysymbol{\glslabel}}{\glslabel}}%
8511          {\glsinsert}}%
8512      }%
8513      {%
```

First use

```
8514      #1{\glsfirstaccessdisplay
8515          {\glentryfirst{\glslabel}}{\glslabel}}%
8516          {\glsdescriptionaccessdisplay
8517              {\glentrydesc{\glslabel}}{\glslabel}}%
8518          {\glssymbolaccessdisplay
8519              {\glentrysymbol{\glslabel}}{\glslabel}}%
8520          {\glsinsert}}%
8521      }%
8522      }%
8523      {%
```

Make first letter upper case

```
8524      \ifglsused\glslabel
8525      {%
```

Subsequent use

```
8526      #2{\glstextaccessdisplay
8527          {\Glsentrytext{\glslabel}}{\glslabel}}%
8528          {\glsdescriptionaccessdisplay
8529              {\glentrydesc{\glslabel}}{\glslabel}}%
8530          {\glssymbolaccessdisplay
8531              {\glentrysymbol{\glslabel}}{\glslabel}}%
8532          {\glsinsert}}%
8533      }%
8534      {%
```

First use

```
8535      #1{\glsfirstaccessdisplay
8536          {\Glsentryfirst{\glslabel}}{\glslabel}}%
8537          {\glsdescriptionaccessdisplay
8538              {\glentrydesc{\glslabel}}{\glslabel}}%
8539          {\glssymbolaccessdisplay
8540              {\glentrysymbol{\glslabel}}{\glslabel}}%
8541          {\glsinsert}}%
8542      }%
```

```

8543     }%
8544     {%

    Make all upper case
8545         \ifglused\glslabel
8546         {%

    Subsequent use
8547         \MakeUppercase{%
8548             #2{\glstextaccessdisplay
8549                 {\glentrytext{\glslabel}}{\glslabel}}%
8550                 {\glsdescriptionaccessdisplay
8551                     {\glentrydesc{\glslabel}}{\glslabel}}%
8552                 {\glssymbolaccessdisplay
8553                     {\glentrysymbol{\glslabel}}{\glslabel}}%
8554                 {\glsinsert}}%
8555         }%
8556         {%

    First use
8557         \MakeUppercase{%
8558             #1{\glsfirstaccessdisplay
8559                 {\glentryfirst{\glslabel}}{\glslabel}}%
8560                 {\glsdescriptionaccessdisplay
8561                     {\glentrydesc{\glslabel}}{\glslabel}}%
8562                 {\glssymbolaccessdisplay
8563                     {\glentrysymbol{\glslabel}}{\glslabel}}%
8564                 {\glsinsert}}%
8565         }%
8566     }%
8567 }%
8568 }%
8569 {%

    Custom text provided in \glsdisp
8570     \ifglused{\glslabel}%
8571     {%

    Subsequent use
8572     #2{\glscustomtext}%
8573     {\glsdescriptionaccessdisplay
8574         {\glentrydesc{\glslabel}}{\glslabel}}%
8575     {\glssymbolaccessdisplay
8576         {\glentrysymbol{\glslabel}}{\glslabel}}%
8577     {\glsinsert}%
8578 }%
8579 {%

    First use
8580     #1{\glscustomtext}%
8581     {\glsdescriptionaccessdisplay
8582         {\glentrydesc{\glslabel}}{\glslabel}}%

```

```

8583      {\glssymbolaccessdisplay
8584      {\glentrysymbol{\glslabel}}{\glslabel}}%
8585      {\glinsert}%
8586    }%
8587  }%
8588 }

```

\@acrshort

```

8589 \def\@acrshort#1#2[#3]{%
8590   \glsdoifexists{#2}%
8591   {%
8592     \edef\@glo@type{\glentrytype{#2}}%

8593     \def\glslabel{#2}%
8594     \let\glsifplural\@secondoftwo
8595     \let\glscapscase\@firstofthree
8596     \let\glinsert\@empty
8597     \def\glscustomtext{%
8598       \acronymfont{\glsshortaccessdisplay{\glentryshort{#2}}{#2}}#3%
8599     }%

    Call \@gls@link
8600     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8601   }%
8602 }

```

\@Acrshort

```

8603 \def\@Acrshort#1#2[#3]{%
8604   \glsdoifexists{#2}%
8605   {%
8606     \edef\@glo@type{\glentrytype{#2}}%

8607     \def\glslabel{#2}%
8608     \let\glsifplural\@secondoftwo
8609     \let\glscapscase\@secondofthree
8610     \let\glinsert\@empty
8611     \def\glscustomtext{%
8612       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
8613     }%

    Call \@gls@link
8614     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8615   }%
8616 }

```

\@ACRshort

```

8617 \def\@ACRshort#1#2[#3]{%
8618   \glsdoifexists{#2}%
8619   {%
8620     \edef\@glo@type{\glentrytype{#2}}%

```

```

8621 \def\glslabel{#2}%
8622 \let\glsifplural\@secondoftwo
8623 \let\glscapscase\@thirdofthree
8624 \let\glsinsert\@empty
8625 \def\glscustomtext{%
8626     \acronymfont{\glsshortaccessdisplay
8627         {\MakeUppercase{\glentryshort{#2}}}{#2}}#3%
8628 }%

Call \@gls@link
8629 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8630 }%
8631 }

```

\@acrlong

```

8632 \def\@acrlong#1#2[#3]{%
8633     \glsdoifexists{#2}%
8634     {%
8635         \edef\@glo@type{\glentrytype{#2}}%

8636         \def\glslabel{#2}%
8637         \let\glsifplural\@secondoftwo
8638         \let\glscapscase\@firstofthree
8639         \let\glsinsert\@empty
8640         \def\glscustomtext{%
8641             \acronymfont{\glslongaccessdisplay{\glentrylong{#2}}{#2}}#3%
8642         }%

Call \@gls@link
8643 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8644 }%
8645 }

```

\@Acrlong

```

8646 \def\@Acrlong#1#2[#3]{%
8647     \glsdoifexists{#2}%
8648     {%
8649         \edef\@glo@type{\glentrytype{#2}}%

8650         \def\glslabel{#2}%
8651         \let\glsifplural\@secondoftwo
8652         \let\glscapscase\@firstofthree
8653         \let\glsinsert\@empty
8654         \def\glscustomtext{%
8655             \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
8656         }%

Call \@gls@link
8657 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8658 }%
8659 }

```

\@ACRlong

```

8660 \def\@ACRlong#1#2[#3]{%
8661   \glsdoifexists{#2}%
8662   {%
8663     \edef\@glo@type{\glentrytype{#2}}%

8664     \def\glslabel{#2}%
8665     \let\glsifplural\@secondoftwo
8666     \let\glscapscase\@firstofthree
8667     \let\glsinsert\@empty
8668     \def\glscustomtext{%
8669       \acronymfont{\glslongaccessdisplay{%
8670         \MakeUppercase{\glentrylong{#2}}}{#2}#3}%
8671     }%

```

Call \@gls@link

```

8672   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8673 }%
8674 }

```

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.

```

8675 \renewcommand*{\glossentryname}[1]{%
8676   \glsdoifexists{#1}%
8677   {%
8678     \glsnamefont{\glsnameaccessdisplay{\glentryname{#1}}{#1}}%
8679   }%
8680 }

8681 \renewcommand*{\glossentryname}[1]{%
8682   \glsdoifexists{#1}%
8683   {%
8684     \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
8685   }%
8686 }

8687 \renewcommand*{\glossentrydesc}[1]{%
8688   \glsdoifexists{#1}%
8689   {%
8690     \glsdescriptionaccessdisplay{\glentrydesc{#1}}{#1}%
8691   }%
8692 }

```



```

8693 \renewcommand*{\Glossentrydesc}[1]{%
8694   \glsdoifexists{#1}%
8695   {%
8696     \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
8697   }%
8698 }

8699 \renewcommand*{\glossentrysymbol}[1]{%
8700   \glsdoifexists{#1}%
8701   {%
8702     \glsymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
8703   }%
8704 }

8705 \renewcommand*{\Glossentrysymbol}[1]{%
8706   \glsdoifexists{#1}%
8707   {%
8708     \glsymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
8709   }%
8710 }

```

pglossaryentryfield

```

8711 \newcommand*{\accsuppglossaryentryfield}[5]{%
8712   \glossaryentryfield{#1}%
8713   {\glsnameaccessdisplay{#2}{#1}}%
8714   {\glsdescriptionaccessdisplay{#3}{#1}}%
8715   {\glsymbolaccessdisplay{#4}{#1}}{#5}%
8716 }

```

glossarysubentryfield

```

8717 \newcommand*{\accsuppglossarysubentryfield}[6]{%
8718   \glossarysubentryfield{#1}{#2}%
8719   {\glsnameaccessdisplay{#3}{#2}}%
8720   {\glsdescriptionaccessdisplay{#4}{#2}}%
8721   {\glsymbolaccessdisplay{#5}{#2}}{#6}%
8722 }

```

6.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```

8723 \renewcommand*{\newacronymhook}{%
8724   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
8725     \the\glskeylisttok}%
8726   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
8727 }

```

DefaultNewAcronymDef Modify default style to use access text:

```

8728 \renewcommand*{\DefaultNewAcronymDef}{%

```

```

8729 \edef\@do@newglossaryentry{%
8730   \noexpand\newglossaryentry{\the\glslabeltok}%
8731   {%
8732     type=\acronymtype,%
8733     name={\the\glsshorttok},%
8734     description={\the\glslongtok},%
8735     descriptionaccess=\relax,
8736     text={\the\glsshorttok},%
8737     access={\noexpand\@glo@textaccess},%
8738     sort={\the\glsshorttok},%
8739     short={\the\glsshorttok},%
8740     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8741     shortaccess={\the\glslongtok},%
8742     long={\the\glslongtok},%
8743     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8744     descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8745     first={\noexpand\glslongaccessdisplay
8746       {\the\glslongtok}{\the\glslabeltok}\space
8747       (\noexpand\glsshortaccessdisplay
8748         {\the\glsshorttok}{\the\glslabeltok})},%
8749     plural={\the\glsshorttok\acrpluralsuffix},%
8750     firstplural={\noexpand\glslongpluralaccessdisplay
8751       {\noexpand\@glo@longpl}{\the\glslabeltok}\space
8752       (\noexpand\glsshortpluralaccessdisplay
8753         {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
8754     firstaccess=\relax,
8755     firstpluralaccess=\relax,
8756     textaccess={\noexpand\@glo@shortaccess},%
8757     \the\glskeylisttok
8758   }%
8759 }%
8760 \@do@newglossaryentry
8761 }

```

otnoteNewAcronymDef

```

8762 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
8763   \edef\@do@newglossaryentry{%
8764     \noexpand\newglossaryentry{\the\glslabeltok}%
8765     {%
8766       type=\acronymtype,%
8767       name={\noexpand\acronymfont{\the\glsshorttok}},%
8768       sort={\the\glsshorttok},%
8769       text={\the\glsshorttok},%
8770       short={\the\glsshorttok},%
8771       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8772       shortaccess={\the\glslongtok},%
8773       long={\the\glslongtok},%
8774       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8775       access={\noexpand\@glo@textaccess},%

```

```

8776 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8777 symbol={\the\glslongtok},%
8778 symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8779 firstpluralaccess=\relax,
8780 textaccess={\noexpand\@glo@shortaccess},%
8781 \the\glskeylisttok
8782 }%
8783 }%
8784 \@do@newglossaryentry
8785 }

```

ptionNewAcronymDef

```

8786 \renewcommand*{\DescriptionNewAcronymDef}{%
8787 \edef\@do@newglossaryentry{%
8788 \noexpand\newglossaryentry{\the\glslabeltok}%
8789 {%
8790 type=\acronymtype,%
8791 name={\noexpand
8792 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
8793 access={\noexpand\@glo@textaccess},%
8794 sort={\the\glsshorttok},%
8795 short={\the\glsshorttok},%
8796 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8797 shortaccess={\the\glslongtok},%
8798 long={\the\glslongtok},%
8799 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8800 first={\the\glslongtok},%
8801 firstaccess=\relax,
8802 firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8803 text={\the\glsshorttok},%
8804 textaccess={\the\glslongtok},%
8805 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8806 symbol={\noexpand\@glo@text},%
8807 symbolaccess={\noexpand\@glo@textaccess},%
8808 symbolplural={\noexpand\@glo@plural},%
8809 firstpluralaccess=\relax,
8810 textaccess={\noexpand\@glo@shortaccess},%
8811 \the\glskeylisttok}%
8812 }%
8813 \@do@newglossaryentry
8814 }

```

otnoteNewAcronymDef

```

8815 \renewcommand*{\FootnoteNewAcronymDef}{%
8816 \edef\@do@newglossaryentry{%
8817 \noexpand\newglossaryentry{\the\glslabeltok}%
8818 {%
8819 type=\acronymtype,%
8820 name={\noexpand\acronymfont{\the\glsshorttok}},%

```

```

8821     sort={\the\glssshorttok},%
8822     text={\the\glssshorttok},%
8823     textaccess={\the\glslongtok},%
8824     access={\noexpand\@glo@textaccess},%
8825     plural={\the\glssshorttok\noexpand\acrpluralsuffix},%
8826     short={\the\glssshorttok},%
8827     shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
8828     long={\the\glslongtok},%
8829     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8830     description={\the\glslongtok},%
8831     descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8832     \the\glskeylisttok
8833 }%
8834 }%
8835 \@do@newglossaryentry
8836 }

```

\SmallNewAcronymDef

```

8837 \renewcommand*{\SmallNewAcronymDef}{%
8838   \edef\@do@newglossaryentry{%
8839     \noexpand\newglossaryentry{\the\glslabeltok}%
8840     {%
8841       type=\acronymtype,%
8842       name={\noexpand\acronymfont{\the\glssshorttok}},%
8843       access={\noexpand\@glo@symbolaccess},%
8844       sort={\the\glssshorttok},%
8845       short={\the\glssshorttok},%
8846       shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
8847       shortaccess={\the\glslongtok},%
8848       long={\the\glslongtok},%
8849       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8850       text={\noexpand\@glo@short},%
8851       textaccess={\noexpand\@glo@shortaccess},%
8852       plural={\noexpand\@glo@shortpl},%
8853       first={\the\glslongtok},%
8854       firstaccess=\relax,%
8855       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8856       description={\noexpand\@glo@first},%
8857       descriptionplural={\noexpand\@glo@firstplural},%
8858       symbol={\the\glssshorttok},%
8859       symbolaccess={\the\glslongtok},%
8860       symbolplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
8861       \the\glskeylisttok
8862     }%
8863   }%
8864   \@do@newglossaryentry
8865 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
8866 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
8867 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
8868 \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
8869 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

6.5 Debugging Commands

```

\showglonameaccess
8870 \newcommand*{\showglonameaccess}[1]{%
8871 \expandafter\show\csname glo@#1@textaccess\endcsname
8872 }

\showglotextaccess
8873 \newcommand*{\showglotextaccess}[1]{%
8874 \expandafter\show\csname glo@#1@textaccess\endcsname
8875 }

showglopluralaccess
8876 \newcommand*{\showglopluralaccess}[1]{%
8877 \expandafter\show\csname glo@#1@pluralaccess\endcsname
8878 }

\showglofirstaccess
8879 \newcommand*{\showglofirstaccess}[1]{%
8880 \expandafter\show\csname glo@#1@firstaccess\endcsname
8881 }

lofirstpluralaccess
8882 \newcommand*{\showglofirstpluralaccess}[1]{%
8883 \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
8884 }

showglosymbolaccess
8885 \newcommand*{\showglosymbolaccess}[1]{%
8886 \expandafter\show\csname glo@#1@symbolaccess\endcsname
8887 }

osymbolpluralaccess
8888 \newcommand*{\showglosymbolpluralaccess}[1]{%
8889 \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
8890 }

```

```

\showglodescaccess
8891 \newcommand*{\showglodescaccess}[1]{%
8892   \expandafter\show\csname glo@#1@descaccess\endcsname
8893 }

glodescpluralaccess
8894 \newcommand*{\showglodescpluralaccess}[1]{%
8895   \expandafter\show\csname glo@#1@descpluralaccess\endcsname
8896 }

\showgloshortaccess
8897 \newcommand*{\showgloshortaccess}[1]{%
8898   \expandafter\show\csname glo@#1@shortaccess\endcsname
8899 }

loshortpluralaccess
8900 \newcommand*{\showgloshortpluralaccess}[1]{%
8901   \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
8902 }

\showglolongaccess
8903 \newcommand*{\showglolongaccess}[1]{%
8904   \expandafter\show\csname glo@#1@longaccess\endcsname
8905 }

glolongpluralaccess
8906 \newcommand*{\showglolongpluralaccess}[1]{%
8907   \expandafter\show\csname glo@#1@longpluralaccess\endcsname
8908 }

```

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.

```

8909 \NeedsTeXFormat{LaTeX2e}
8910 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]

English:
8911 \@ifundefined{captionsenglish}{}{%
8912   \addto\captionsenglish{%
8913     \renewcommand*{\glossaryname}{Glossary}%
8914     \renewcommand*{\acronymname}{Acronyms}%

```

```

8915 \renewcommand*{\entryname}{Notation}%
8916 \renewcommand*{\descriptionname}{Description}%
8917 \renewcommand*{\symbolname}{Symbol}%
8918 \renewcommand*{\pagelistname}{Page List}%
8919 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8920 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8921 }%
8922 }
8923 \@ifundefined{captionsamerican}{}{%
8924 \addto\captionsamerican{%
8925 \renewcommand*{\glossaryname}{Glossary}%
8926 \renewcommand*{\acronymname}{Acronyms}%
8927 \renewcommand*{\entryname}{Notation}%
8928 \renewcommand*{\descriptionname}{Description}%
8929 \renewcommand*{\symbolname}{Symbol}%
8930 \renewcommand*{\pagelistname}{Page List}%
8931 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8932 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8933 }%
8934 }
8935 \@ifundefined{captionsaustralian}{}{%
8936 \addto\captionsaustralian{%
8937 \renewcommand*{\glossaryname}{Glossary}%
8938 \renewcommand*{\acronymname}{Acronyms}%
8939 \renewcommand*{\entryname}{Notation}%
8940 \renewcommand*{\descriptionname}{Description}%
8941 \renewcommand*{\symbolname}{Symbol}%
8942 \renewcommand*{\pagelistname}{Page List}%
8943 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8944 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8945 }%
8946 }
8947 \@ifundefined{captionsbritish}{}{%
8948 \addto\captionsbritish{%
8949 \renewcommand*{\glossaryname}{Glossary}%
8950 \renewcommand*{\acronymname}{Acronyms}%
8951 \renewcommand*{\entryname}{Notation}%
8952 \renewcommand*{\descriptionname}{Description}%
8953 \renewcommand*{\symbolname}{Symbol}%
8954 \renewcommand*{\pagelistname}{Page List}%
8955 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8956 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8957 }%
8958 \@ifundefined{captionscanadian}{}{%
8959 \addto\captionscanadian{%
8960 \renewcommand*{\glossaryname}{Glossary}%
8961 \renewcommand*{\acronymname}{Acronyms}%
8962 \renewcommand*{\entryname}{Notation}%
8963 \renewcommand*{\descriptionname}{Description}%

```

```

8964 \renewcommand*{\symbolname}{Symbol}%
8965 \renewcommand*{\pagelistname}{Page List}%
8966 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8967 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8968 }%
8969 }
8970 \@ifundefined{captionsnewzealand}{}{%
8971 \addto\captionsnewzealand{%
8972 \renewcommand*{\glossaryname}{Glossary}%
8973 \renewcommand*{\acronymname}{Acronyms}%
8974 \renewcommand*{\entryname}{Notation}%
8975 \renewcommand*{\descriptionname}{Description}%
8976 \renewcommand*{\symbolname}{Symbol}%
8977 \renewcommand*{\pagelistname}{Page List}%
8978 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8979 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8980 }%
8981 }
8982 \@ifundefined{captionsUKenglish}{}{%
8983 \addto\captionsUKenglish{%
8984 \renewcommand*{\glossaryname}{Glossary}%
8985 \renewcommand*{\acronymname}{Acronyms}%
8986 \renewcommand*{\entryname}{Notation}%
8987 \renewcommand*{\descriptionname}{Description}%
8988 \renewcommand*{\symbolname}{Symbol}%
8989 \renewcommand*{\pagelistname}{Page List}%
8990 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8991 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8992 }%
8993 }
8994 \@ifundefined{captionsUSenglish}{}{%
8995 \addto\captionsUSenglish{%
8996 \renewcommand*{\glossaryname}{Glossary}%
8997 \renewcommand*{\acronymname}{Acronyms}%
8998 \renewcommand*{\entryname}{Notation}%
8999 \renewcommand*{\descriptionname}{Description}%
9000 \renewcommand*{\symbolname}{Symbol}%
9001 \renewcommand*{\pagelistname}{Page List}%
9002 \renewcommand*{\glssymbolsgroupname}{Symbols}%
9003 \renewcommand*{\glsnumbersgroupname}{Numbers}%
9004 }%
9005 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

9006 \@ifundefined{captionsgerman}{}{%
9007 \addto\captionsgerman{%
9008 \renewcommand*{\glossaryname}{Glossar}%
9009 \renewcommand*{\acronymname}{Akronyme}%
9010 \renewcommand*{\entryname}{Bezeichnung}%

```



```

9011 \renewcommand*{\descriptionname}{Beschreibung}%
9012 \renewcommand*{\symbolname}{Symbol}%
9013 \renewcommand*{\pagelistname}{Seiten}%
9014 \renewcommand*{\glssymbolsgroupname}{Symbole}%
9015 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
9016 }

```

ngerman is identical to German:

```

9017 \@ifundefined{captionsngerman}{}{%
9018 \addto\captionsngerman{%
9019 \renewcommand*{\glossaryname}{Glossar}%
9020 \renewcommand*{\acronymname}{Akronyme}%
9021 \renewcommand*{\entryname}{Bezeichnung}%
9022 \renewcommand*{\descriptionname}{Beschreibung}%
9023 \renewcommand*{\symbolname}{Symbol}%
9024 \renewcommand*{\pagelistname}{Seiten}%
9025 \renewcommand*{\glssymbolsgroupname}{Symbole}%
9026 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
9027 }

```

Italian:

```

9028 \@ifundefined{captionsitalian}{}{%
9029 \addto\captionsitalian{%
9030 \renewcommand*{\glossaryname}{Glossario}%
9031 \renewcommand*{\acronymname}{Acronimi}%
9032 \renewcommand*{\entryname}{Nomenclatura}%
9033 \renewcommand*{\descriptionname}{Descrizione}%
9034 \renewcommand*{\symbolname}{Simbolo}%
9035 \renewcommand*{\pagelistname}{Elenco delle pagine}%
9036 \renewcommand*{\glssymbolsgroupname}{Simboli}%
9037 \renewcommand*{\glsnumbersgroupname}{Numeri}}
9038 }

```

Dutch:

```

9039 \@ifundefined{captionsdutch}{}{%
9040 \addto\captionsdutch{%
9041 \renewcommand*{\glossaryname}{Woordenlijst}%
9042 \renewcommand*{\acronymname}{Acroniemen}%
9043 \renewcommand*{\entryname}{Benaming}%
9044 \renewcommand*{\descriptionname}{Beschrijving}%
9045 \renewcommand*{\symbolname}{Symbool}%
9046 \renewcommand*{\pagelistname}{Pagina's}%
9047 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
9048 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
9049 }

```

Spanish:

```

9050 \@ifundefined{captionsspanish}{}{%
9051 \addto\captionsspanish{%
9052 \renewcommand*{\glossaryname}{Glosario}%
9053 \renewcommand*{\acronymname}{Siglas}%

```

```

9054 \renewcommand*{\entryname}{Entrada}%
9055 \renewcommand*{\descriptionname}{Descripci\`on}%
9056 \renewcommand*{\symbolname}{S\`{\i}mbolo}%
9057 \renewcommand*{\pagelistname}{Lista de p\`aginas}%
9058 \renewcommand*{\glssymbolsgroupname}{S\`{\i}mbolos}%
9059 \renewcommand*{\glslnumbersgroupname}{N\`umeros}}
9060 }

```

French:

```

9061 \@ifundefined{captionsfrench}{}{%
9062 \addto\captionsfrench{%
9063 \renewcommand*{\glossaryname}{Glossaire}%
9064 \renewcommand*{\acronymname}{Acronymes}%
9065 \renewcommand*{\entryname}{Terme}%
9066 \renewcommand*{\descriptionname}{Description}%
9067 \renewcommand*{\symbolname}{Symbole}%
9068 \renewcommand*{\pagelistname}{Pages}%
9069 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9070 \renewcommand*{\glslnumbersgroupname}{Nombres}}
9071 }
9072 \@ifundefined{captionsfrenchb}{}{%
9073 \addto\captionsfrenchb{%
9074 \renewcommand*{\glossaryname}{Glossaire}%
9075 \renewcommand*{\acronymname}{Acronymes}%
9076 \renewcommand*{\entryname}{Terme}%
9077 \renewcommand*{\descriptionname}{Description}%
9078 \renewcommand*{\symbolname}{Symbole}%
9079 \renewcommand*{\pagelistname}{Pages}%
9080 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9081 \renewcommand*{\glslnumbersgroupname}{Nombres}}
9082 }
9083 \@ifundefined{captionsfrancais}{}{%
9084 \addto\captionsfrancais{%
9085 \renewcommand*{\glossaryname}{Glossaire}%
9086 \renewcommand*{\acronymname}{Acronymes}%
9087 \renewcommand*{\entryname}{Terme}%
9088 \renewcommand*{\descriptionname}{Description}%
9089 \renewcommand*{\symbolname}{Symbole}%
9090 \renewcommand*{\pagelistname}{Pages}%
9091 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9092 \renewcommand*{\glslnumbersgroupname}{Nombres}}
9093 }

```

Danish:

```

9094 \@ifundefined{captionsdanish}{}{%
9095 \addto\captionsdanish{%
9096 \renewcommand*{\glossaryname}{Ordliste}%
9097 \renewcommand*{\acronymname}{Akronymer}%
9098 \renewcommand*{\entryname}{Symbolforklaring}%
9099 \renewcommand*{\descriptionname}{Beskrivelse}%

```

```

9100 \renewcommand*{\symbolname}{Symbol}%
9101 \renewcommand*{\pagelistname}{Side}%
9102 \renewcommand*{\glssymbolsgroupname}{Symboler}%
9103 \renewcommand*{\glsnumbersgroupname}{Tal}%
9104 }

```

Irish:

```

9105 \@ifundefined{captionsirish}{}{%
9106 \addto\captionsirish{%
9107 \renewcommand*{\glossaryname}{Gluais}%
9108 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

9109 \renewcommand*{\entryname}{Ciall}%
9110 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

9111 \renewcommand*{\symbolname}{Comhartha}%
9112 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}%
9113 \renewcommand*{\pagelistname}{Leathanaigh}%
9114 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}%
9115 }

```

Hungarian:

```

9116 \@ifundefined{captionsmagyar}{}{%
9117 \addto\captionsmagyar{%
9118 \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
9119 \renewcommand*{\acronymname}{Bet\H uszavak}%
9120 \renewcommand*{\entryname}{Kifejez\'es}%
9121 \renewcommand*{\descriptionname}{Magyar\'azat}%
9122 \renewcommand*{\symbolname}{Jel\'ol\'es}%
9123 \renewcommand*{\pagelistname}{Oldalsz\'am}%
9124 \renewcommand*{\glssymbolsgroupname}{Jelek}%
9125 \renewcommand*{\glsnumbersgroupname}{Sz\'amjegyek}%
9126 }
9127 }
9128 \@ifundefined{captionshungarian}{}{%
9129 \addto\captionshungarian{%
9130 \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
9131 \renewcommand*{\acronymname}{Bet\H uszavak}%
9132 \renewcommand*{\entryname}{Kifejez\'es}%
9133 \renewcommand*{\descriptionname}{Magyar\'azat}%
9134 \renewcommand*{\symbolname}{Jel\'ol\'es}%
9135 \renewcommand*{\pagelistname}{Oldalsz\'am}%
9136 \renewcommand*{\glssymbolsgroupname}{Jelek}%
9137 \renewcommand*{\glsnumbersgroupname}{Sz\'amjegyek}%
9138 }
9139 }

```

Polish

```

9140 \@ifundefined{captionspolish}{}{%
9141   \addto\captionspolish{%
9142     \renewcommand*{\glossaryname}{S{\l}ownik termin\'}ow}%
9143     \renewcommand*{\acronymname}{Skr\'}ot}%
9144     \renewcommand*{\entryname}{Termin}%
9145     \renewcommand*{\descriptionname}{Opis}%
9146     \renewcommand*{\symbolname}{Symbol}%
9147     \renewcommand*{\pagelistname}{Strony}%
9148     \renewcommand*{\glssymbolsgroupname}{Symbole}%
9149     \renewcommand*{\glsnumbersgroupname}{Liczby}}
9150 }

```

Brazilian

```

9151 \@ifundefined{captionsbrazil}{}{%
9152   \addto\captionsbrazil{%
9153     \renewcommand*{\glossaryname}{Gloss\'}ario}%
9154     \renewcommand*{\acronymname}{Siglas}%
9155     \renewcommand*{\entryname}{Nota\ c\ ~ao}%
9156     \renewcommand*{\descriptionname}{Descri\ c\ ~ao}%
9157     \renewcommand*{\symbolname}{S\'}imbolo}%
9158     \renewcommand*{\pagelistname}{Lista de P\'}aginas}%
9159     \renewcommand*{\glssymbolsgroupname}{S\'}imbolos}%
9160     \renewcommand*{\glsnumbersgroupname}{N\'}umeros}%
9161   }%
9162 }

```

7.2 Polyglossia Captions

```

9163 \NeedsTeXFormat{LaTeX2e}
9164 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

English:

```

9165 \@ifundefined{captionseenglish}{}{%
9166   \expandafter\toks@\expandafter{\captionseenglish
9167     \renewcommand*{\glossaryname}{\textenglish{Glossary}}}%
9168     \renewcommand*{\acronymname}{\textenglish{Acronyms}}}%
9169     \renewcommand*{\entryname}{\textenglish{Notation}}}%
9170     \renewcommand*{\descriptionname}{\textenglish{Description}}}%
9171     \renewcommand*{\symbolname}{\textenglish{Symbol}}}%
9172     \renewcommand*{\pagelistname}{\textenglish{Page List}}}%
9173     \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}}%
9174     \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}}%
9175   }%
9176   \edef\captionseenglish{\the\toks@}%
9177 }

```

German:

```

9178 \@ifundefined{captionsgerman}{}{%
9179   \expandafter\toks@\expandafter{\captionsgerman
9180     \renewcommand*{\glossaryname}{\textgerman{Glossar}}}%

```

```

9181 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
9182 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
9183 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
9184 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
9185 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
9186 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
9187 \renewcommand*{\glsnumpersgroupname}{\textgerman{Zahlen}}%
9188 }%
9189 \edef\captionsgerman{\the\toks@}%
9190 }

Italian:
9191 \@ifundefined{captionsspanish}{\textspanish{Glosario}}%
9192 \expandafter\toks@\expandafter{\captionsspanish
9193 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
9194 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
9195 \renewcommand*{\entryname}{\textspanish{Entrada}}%
9196 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
9197 \renewcommand*{\symbolname}{\textspanish{S' \i mbolo}}%
9198 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
9199 \renewcommand*{\glssymbolsgroupname}{\textspanish{S' \i mbolos}}%
9200 \renewcommand*{\glsnumpersgroupname}{\textspanish{Numeri}}%
9201 }%
9202 \edef\captionsspanish{\the\toks@}%
9203 }

Dutch:
9204 \@ifundefined{captionsspanish}{\textspanish{Glosario}}%
9205 \expandafter\toks@\expandafter{\captionsspanish
9206 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
9207 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
9208 \renewcommand*{\entryname}{\textspanish{Entrada}}%
9209 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
9210 \renewcommand*{\symbolname}{\textspanish{S' \i mbolo}}%
9211 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
9212 \renewcommand*{\glssymbolsgroupname}{\textspanish{S' \i mbolos}}%
9213 \renewcommand*{\glsnumpersgroupname}{\textspanish{Numeri}}%
9214 }%
9215 \edef\captionsspanish{\the\toks@}%
9216 }

Spanish:
9217 \@ifundefined{captionsspanish}{\textspanish{Glosario}}%
9218 \expandafter\toks@\expandafter{\captionsspanish
9219 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
9220 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
9221 \renewcommand*{\entryname}{\textspanish{Entrada}}%
9222 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
9223 \renewcommand*{\symbolname}{\textspanish{S' \i mbolo}}%
9224 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
9225 \renewcommand*{\glssymbolsgroupname}{\textspanish{S' \i mbolos}}%

```

```

9226 \renewcommand*{\glnumbersgroupname}{\textspanish{N\'umeros}}%
9227 }%
9228 \edef\captionsspanish{\the\toks@}%
9229 }

```

French:

```

9230 \@ifundefined{captionsfrench}{}{%
9231 \expandafter\toks@\expandafter{\captionsfrench
9232 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
9233 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
9234 \renewcommand*{\entryname}{\textfrench{Terme}}%
9235 \renewcommand*{\descriptionname}{\textfrench{Description}}%
9236 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
9237 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
9238 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
9239 \renewcommand*{\glnumbersgroupname}{\textfrench{Nombres}}%
9240 }%
9241 \edef\captionsfrench{\the\toks@}%
9242 }

```

Danish:

```

9243 \@ifundefined{captionsdanish}{}{%
9244 \expandafter\toks@\expandafter{\captionsdanish
9245 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
9246 \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
9247 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
9248 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
9249 \renewcommand*{\symbolname}{\textdanish{Symbol}}%
9250 \renewcommand*{\pagelistname}{\textdanish{Side}}%
9251 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
9252 \renewcommand*{\glnumbersgroupname}{\textdanish{Tal}}%
9253 }%
9254 \edef\captionsdanish{\the\toks@}%
9255 }

```

Irish:

```

9256 \@ifundefined{captionsirish}{}{%
9257 \expandafter\toks@\expandafter{\captionsirish
9258 \renewcommand*{\glossaryname}{\textirish{Gluais}}%
9259 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%
9260 \renewcommand*{\entryname}{\textirish{Ciall}}%
9261 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}%
9262 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
9263 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\'{\i}}}%
9264 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%
9265 \renewcommand*{\glnumbersgroupname}{\textirish{Uimhreacha}}%
9266 }%
9267 \edef\captionsirish{\the\toks@}%
9268 }

```

Hungarian:

```

9269 \@ifundefined{captionsmagyar}{}{%

```

```

9270 \expandafter\toks@\expandafter{\captionsmagyar
9271 \renewcommand*{\glossaryname}{\textmagyar{Sz\'ojegyz\'ek}}}%
9272 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
9273 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}}%
9274 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'azat}}}%
9275 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}}%
9276 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}}%
9277 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
9278 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\'amjegyek}}}%
9279 }%
9280 \edef\captionsmagyar{\the\toks@}%
9281 }

Polish
9282 \@ifundefined{captionspolish}{}{%
9283 \expandafter\toks@\expandafter{\captionspolish
9284 \renewcommand*{\glossaryname}{\textpolish{S\lownik termin\'ow}}}%
9285 \renewcommand*{\acronymname}{\textpolish{Skr\'ot}}}%
9286 \renewcommand*{\entryname}{\textpolish{Termin}}}%
9287 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
9288 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
9289 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
9290 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%
9291 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}}%
9292 }%
9293 \edef\captionspolish{\the\toks@}%
9294 }

Portugues
9295 \@ifundefined{captionsportuges}{}{%
9296 \expandafter\toks@\expandafter{\captionsportuges
9297 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}}%
9298 \renewcommand*{\acronymname}{\textportuges{Siglas}}}%
9299 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}}%
9300 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}}%
9301 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}}%
9302 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}}%
9303 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}}%
9304 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}}%
9305 }%
9306 \edef\captionsportuges{\the\toks@}%
9307 }

```

7.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```
9308 \ProvidesDictionary{glossaries-dictionary}{Brazilian}
```

Provide Brazilian translations:

```

9309 \providetranslation{Glossary}{Gloss\'ario}
9310 \providetranslation{Acronyms}{Siglas}
9311 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}

```

```

9312 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
9313 \providetranslation{Symbol (glossaries)}{S\'imbolo}
9314 \providetranslation{Page List (glossaries)}{Lista de P\'aginas}
9315 \providetranslation{Symbols (glossaries)}{S\'imbolos}
9316 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

7.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```
9317 \ProvidesDictionary{glossaries-dictionary}{Danish}
```

Provide Danish translations:

```

9318 \providetranslation{Glossary}{Ordliste}
9319 \providetranslation{Acronyms}{Akronymer}
9320 \providetranslation{Notation (glossaries)}{Symbolforklaring}
9321 \providetranslation{Description (glossaries)}{Beskrivelse}
9322 \providetranslation{Symbol (glossaries)}{Symbol}
9323 \providetranslation{Page List (glossaries)}{Side}
9324 \providetranslation{Symbols (glossaries)}{Symboler}
9325 \providetranslation{Numbers (glossaries)}{Tal}

```

7.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
9326 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```

9327 \providetranslation{Glossary}{Woordenlijst}
9328 \providetranslation{Acronyms}{Acroniemen}
9329 \providetranslation{Notation (glossaries)}{Benaming}
9330 \providetranslation{Description (glossaries)}{Beschrijving}
9331 \providetranslation{Symbol (glossaries)}{Symbool}
9332 \providetranslation{Page List (glossaries)}{Pagina's}
9333 \providetranslation{Symbols (glossaries)}{Symbolen}
9334 \providetranslation{Numbers (glossaries)}{Cijfers}

```

7.6 English Dictionary

This is a dictionary file provided for use with the package.

```
9335 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```

9336 \providetranslation{Glossary}{Glossary}
9337 \providetranslation{Acronyms}{Acronyms}
9338 \providetranslation{Notation (glossaries)}{Notation}
9339 \providetranslation{Description (glossaries)}{Description}
9340 \providetranslation{Symbol (glossaries)}{Symbol}
9341 \providetranslation{Page List (glossaries)}{Page List}
9342 \providetranslation{Symbols (glossaries)}{Symbols}
9343 \providetranslation{Numbers (glossaries)}{Numbers}

```


7.7 French Dictionary

This is a dictionary file provided for use with the package.

```
9344 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
9345 \providetranslation{Glossary}{Glossaire}
9346 \providetranslation{Acronyms}{Acronymes}
9347 \providetranslation{Notation (glossaries)}{Terme}
9348 \providetranslation{Description (glossaries)}{Description}
9349 \providetranslation{Symbol (glossaries)}{Symbole}
9350 \providetranslation{Page List (glossaries)}{Pages}
9351 \providetranslation{Symbols (glossaries)}{Symboles}
9352 \providetranslation{Numbers (glossaries)}{Nombres}
```

7.8 German Dictionary

This is a dictionary file provided for use with the package.

```
9353 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
9354 \providetranslation{Glossary}{Glossar}
9355 \providetranslation{Acronyms}{Akronyme}
9356 \providetranslation{Notation (glossaries)}{Bezeichnung}
9357 \providetranslation{Description (glossaries)}{Beschreibung}
9358 \providetranslation{Symbol (glossaries)}{Symbol}
9359 \providetranslation{Page List (glossaries)}{Seiten}
9360 \providetranslation{Symbols (glossaries)}{Symbole}
9361 \providetranslation{Numbers (glossaries)}{Zahlen}
```

7.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
9362 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
9363 \providetranslation{Glossary}{Gluais}
9364 \providetranslation{Acronyms}{Acrainmneacha}
9365 \providetranslation{Notation (glossaries)}{Ciall}
9366 \providetranslation{Description (glossaries)}{Tuairisc}
9367 \providetranslation{Symbol (glossaries)}{Comhartha}
9368 \providetranslation{Page List (glossaries)}{Leathanaigh}
9369 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
9370 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

7.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
9371 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
9372 \providetranslation{Glossary}{Glossario}
9373 \providetranslation{Acronyms}{Acronimi}
9374 \providetranslation{Notation (glossaries)}{Nomenclatura}
9375 \providetranslation{Description (glossaries)}{Descrizione}
9376 \providetranslation{Symbol (glossaries)}{Simbolo}
9377 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
9378 \providetranslation{Symbols (glossaries)}{Simboli}
9379 \providetranslation{Numbers (glossaries)}{Numeri}
```

7.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
9380 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
9381 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
9382 \providetranslation{Acronyms}{Bet\'H uszavak}
9383 \providetranslation{Notation (glossaries)}{Kifejez\'es}
9384 \providetranslation{Description (glossaries)}{Magyar\'azat}
9385 \providetranslation{Symbol (glossaries)}{Jel\'ol\'es}
9386 \providetranslation{Page List (glossaries)}{Oldalsz\'am}
9387 \providetranslation{Symbols (glossaries)}{Jelek}
9388 \providetranslation{Numbers (glossaries)}{Sz\'amjegyek}
```

7.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
9389 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```
9390 \providetranslation{Glossary}{S{\l}ownik termin\'ow}
9391 \providetranslation{Acronyms}{Skr\'ot}
9392 \providetranslation{Notation (glossaries)}{Termin}
9393 \providetranslation{Description (glossaries)}{Opis}
9394 \providetranslation{Symbol (glossaries)}{Symbol}
9395 \providetranslation{Page List (glossaries)}{Strony}
9396 \providetranslation{Symbols (glossaries)}{Symbole}
9397 \providetranslation{Numbers (glossaries)}{Liczby}
```

7.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```
9398 \ProvidesDictionary{glossaries-dictionary}{Serbian}
9399 \providetranslation{Glossary}{Mali re\'v cnik}
9400 \providetranslation{Acronyms}{Skra\' cenice}
9401 \providetranslation{Notation (glossaries)}{Oznaka}
9402 \providetranslation{Description (glossaries)}{Opis}
9403 \providetranslation{Symbol (glossaries)}{Simbol}
```

```

9404 \providetranslation{Page List (glossaries)}{Stranica}
9405 \providetranslation{Symbols (glossaries)}{Simboli}
9406 \providetranslation{Numbers (glossaries)}{Brojevi}

```

7.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

9407 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

9408 \providetranslation{Glossary}{Glosario}
9409 \providetranslation{Acronyms}{Siglas}
9410 \providetranslation{Notation (glossaries)}{Entrada}
9411 \providetranslation{Description (glossaries)}{Descripci\on}
9412 \providetranslation{Symbol (glossaries)}{S'\i mbolo}
9413 \providetranslation{Page List (glossaries)}{Lista de p\ aginas}
9414 \providetranslation{Symbols (glossaries)}{S'\i mbolos}
9415 \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 83	Changed the default value of the sort key to just the value of the name key 63
	<code>\writeist</code> : Added spaces after <code>\delimN</code> and <code>\delimR</code> in ist file 140	<code>\glsmakefirstuc</code> : new 208
1.03	<code>\makefirstuc</code> : changed 'protected@edef' to 'def' 208	1.06 General: now requires <code>etoolbox</code> . 207 <code>\capitalisewords</code> : new 209 <code>\xcapitalisewords</code> : new 209
1.04	General: Added <code>\glstextformat</code> 72	1.07 <code>\@gls@link</code> : fixed bug caused by <code>\theglscopycounter</code> setting the page number too soon 82
1.05	<code>\glossarysection</code> : added <code>\@mkboth</code> to <code>\glossarysection</code> 32	<code>\glsadd</code> : fixed bug caused by <code>\theglscopycounter</code> setting the page number too soon 138
	<code>\gls@defglossaryentry</code> :	

- 1.08
- General: Added babel support ... 26
 - \capitalisewords: made robust 209
 - listgroup: changed listgroup style to use \glsgetgrouptitle 215
 - altlistgroup: changed altlistgroup style to use \glsgetgrouptitle 216
 - \makefirstuc: made robust ... 207
- 1.1
- \@glossarysection: numbered sections and auto label added 33
 - \@gls@tmpb: changed \toksdef to \newtoks 86
 - \@gls@toc: numberline added .. 34
 - \@p@glossarysection: numbered sections and auto label added 33
 - General: Added support for translator package 27
 - amsgen now loaded (\new@ifnextchar needed) 4
 - translate: translate option added 19
 - \setglossarysection: new ... 33
 - 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 100
 - \@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol 98
 - General: added check for \hypertarget separate to \hyperlink (memoir defines \hyperlink but not \hypertarget) 91
 - descriptionplural: new 50
 - \gls@defglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended) 63
 - descriptionplural support added 63
 - symbolplural support added .. 63
 - \Glsentrydescplural: New .. 132
 - \glsentrydescplural: New .. 132
 - \Glsentrysymbolplural: New 133
 - \glsentrysymbolplural: New 133
 - \SetDescriptionFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 180
 - \SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 186
 - symbolplural: new 51
- 1.13
- General: Add Polish support 300, 303
 - fixed bug that ignored 3rd parameter 102–115
 - \ACRfullpl: new 175
 - \Acrfullpl: new 175
 - \acrfullpl: new 175
 - \acrpluralsuffix: New 173
 - \gls@defglossaryentry: Changed default first value .. 63
 - Changed default firstplural value 63
 - Removed restriction on only using \newglossaryentry in the preamble 67
 - \newacronym: Removed restriction on only using \newacronym in the preamble 173
- 1.14
- \@gls@hypergroup: new 211
 - General: added nonnumberlist key to \printglossary 159
 - added numberedsection key to \printglossary 159
 - \firstacronymfont: new 176
 - \glsautoprefix: new 6
 - \glsnavhyperlink: changed 'edef to 'protected@edef ... 210
 - \glsnavhypertarget: added write to aux file 210
 - \glsnavigation: changed to only use labels for groups that are present 211

1.15	1.17
\@gls@link: added \glslabel . 82	\@@do@wrglossary: new 151
General: Added \glssettoctitle	\@do@seeglossary: new 153
..... 27	\@glo@storeentry: new 68
\gls@defglossaryentry: check	\@glossary: changed defini-
for \@glo@first in descrip-	tion to use \index instead of
tion 66	\@index 149
check for \@glo@text in sym-	\@glsdefaultplural: new 54
bol 67	\@glsdefaultsort: new 54
\gls@hypergroup: new . 210	\@gls@hypernumber: new 169
\glsnavhypertarget: added	\@glsnoname: new 54
check if rerun required 210	\@glsnonextpages: new 159
\glssettoctitle: new 26	\@wrglossary: modified to allow
\printglossary: changed the	for xindy support 150
way the TOC title is set 155	General: added Brazilian dictio-
1.16	nary 303
\@GLS@: Test glossary type is	Added Brazilian support 300
\acronymtype in addition to	added xindy support 21
checking if footnote option has	parent: new 52
been used 96	see: new 52
\@GLSpl: Test glossary type is	\gls@defglossaryentry: added
\acronymtype in addition to	nonumberlist key 63
checking if footnote option has	added parent key 63
been used 100	added see key 63
\@Gls@: Test glossary type is	Stored main part of entry format
\acronymtype in addition to	when entry is defined 67
checking if footnote option has	\gls@suffixF: new 30
been used 95	\gls@suffixFF: new 30
\@Glspl@: Test glossary type is	\gls@hyperlink: new 137
\acronymtype in addition to	\gls@hypernumber: modified to
checking if footnote option has	allow material to be attached
been used 99	to location 169
\@Gls@: Test glossary type is	\glsnavhyperlink: replaced 'hy-
\acronymtype in addition to	perlink to '@glslink 210
checking if footnote option has	\glsnavhypertarget: replaced
been used 93	'hypertarget to '@glstarget . 210
\@Glsdisp: Test glossary type is	\glssee: new 154
\acronymtype in addition to	\glsseeformat: new 154
checking if footnote option has	\glsSetSuffixF: new 30
been used 101	\glsSetSuffixFF: new 30
\@Glspl@: Test glossary type is	\ifglsxindy: new 21
\acronymtype in addition to	\istfilename: added xindy sup-
checking if footnote option has	port 29
been used 97	\newglossarystyle: made
\@glstarget: raised the hyper-	\newglossarystyle long . 168
target so the target text doesn't	\nopostdesc: new 28
scroll off the top of the page . 91	nonumberlist: new 52
\gls@defglossaryentry:	
Changed def to let 63	

\printglossary: added check to determine if \printglossary is already defined 155 added print language to aux file 155 order: order package option added 21 \writeist: added xindy support 140	2.01	\@gls@link: moved \@do@wrglossary before term is displayed to pre- vent unwanted whatsit 82 \forallglossaries: replaced \ifthenelse with \ifx 44 \forglseentries: replaced \ifthenelse with \ifx 44 \glsldefmain: new 12 \glsldescwidth: changed \linewidth to \hspace . 217, 233 \glslistdottedwidth: changed \linewidth to \hspace 217 \glspagelistwidth: changed \linewidth to \hspace . 218, 233 nomain: added nomain package option 12 \writeist: removed item_02 - no such makeindex key 144
1.18 \@gls@loadlist: new 8 \@gls@loadlong: new 7 \@gls@loadsuper: new 8 \@gls@loadtree: new 8 \glsldefglossaryentry: Changed default value of sort to \@glsldefaultsort 63 moved sort sanitization to \newglossaryentry 67 \glstarget: new 163 \oldacronym: new 172 nolist: new 8 nolong: new 8 sort: moved sanitization to \newglossaryentry 50 nostyles: new 8 nosuper: new 8 notree: new 8	2.02	General: Changed Brazil to Brazil- ian 303 false will prevent automatic loading of translator package 24 \glossarysection: changed \@mkboth to \glossarymark 32 \glsglossarymark: New 32 \printglossary: suppressed warning globally rather than locally 158
1.19 \glsclearpage: new 34 \glldisp: new 100 \SetDescriptionAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller 184 \SetDescriptionFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller 180 \SetFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller 186 \SetSmallAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller 189	2.03	\@GLS@: Added check for hyper- first 96 \@GLSpl: Added check for hyper- first 100 \@Gls@: Added check for hyper- first 95 \@Glspl@: Added check for hyper- first 99 \@gls@: Added check for hyper- first 93 \@gls@link: new 81 \@gls@link: added \leavevmode 82 Moved entry existence check to avoid duplicate code 82 \@glldisp: Added check for hy- perfirst 101
1.2 General: fixed bug in ngerman captions 297		

\@glsp1@: Added check for hyper-first	97	\glentryuseriv: new	134
\glsglossarymark: Added check to see if it's already defined ..	32	\Glsentryuseriv: new	135
hyperfirst: new	20	\glentryuseriv: new	134
2.04		\Glsentryuseriv: new	135
\@GLS@: Changed test to check if glossary type has been identified as a list of acronyms	96	\glentryuseriv: new	135
\@GLSp1: Changed test to check if glossary type has been identified as a list of acronyms ...	100	\newglossary: added check to determine if \@glS@<type>@display and \@glS@<type>@displayfirst have been defined.	48
\@GLs@: Changed test to check if glossary type has been identified as a list of acronyms	95	\SetAcronymLists: new	15
\@GLsp1@: Changed test to check if glossary type has been identified as a list of acronyms ...	99	\SetDefaultAcronymDisplayStyle: new	176
\@glossaryentryfield: new ..	68	\SetDefaultAcronymStyle: new	177
\@glossarysubentryfield: new	68	\SetDescriptionAcronymDisplayStyle: new	182
\@glS@: Changed test to check if glossary type has been identified as a list of acronyms	93	\SetDescriptionDUAAcronymDisplayStyle: new	180
\@glSacronymlists: new	13	\SetDescriptionFootnoteAcronymDisplayStyle: new	178
\@glSdisp: Changed test to check if glossary type has been identified as a list of acronyms ..	101	\SetDUADisplayStyle: new ..	189
\@glsp1@: Changed test to check if glossary type has been identified as a list of acronyms ...	97	\SetFootnoteAcronymDisplayStyle: new	184
\@newglossaryentryposthook: new	67	\SetSmallAcronymDisplayStyle: new	187
\@newglossaryentryprehook: new	67	2.05	
acronymlists: new	15	\@glSdisp: Added closing brace. Patch provided by Sergiu Dotenco	101
\DeclareAcronymList: new ...	14	Removed spurious brace. Patch provided by Sergiu Dotenco	101
\DefineAcronymSynonyms: new	193	\writeist: Added \string before opening and closing braces. Patch provided by Segiu Dotenco	145
\glS@defglossaryentry: added user1-6 keys	63	2.06	
\glSadd: fixed bug that ignored counter	138	\altnewglossary: new	49
\Glsentryuseri: new	134	\CustomAcronymFields: new ..	192
\Glsentryuseri: new	134	\CustomNewAcronymDef: new ..	192
\Glsentryuserii: new	134	\SetCustomDisplayStyle: new	192
\Glsentryuserii: new	134	\SetCustomStyle: new	192
\Glsentryuseriii: new	134	2.07	
\Glsentryuseriii: new	134	General: glssadd format key stored in \@glSnumberformat (was mistakenly stored in \@glo@format)	138
\Glsentryuseriv: new	134	3.0	
		\@do@wrglossary: added check for hyper location prefix ...	151

modified to use new format ..	151	see:added \@glo@seeautonumberlist	
\@glossarysec: replaced		52
\@ifundefined with		seeautonumberlist: new	7
\ifcsundef	5	\glossarysection: replaced	
\@do@seeglossary: Sanitize and		\@ifundefined with	
escape cross-referencing in-		\ifcsundef	32
formation	153	\glossarystyle: replaced	
\@gls@counterwithin: new	9	\@ifundefined with	
\@gls@ifinlist: new	35	\ifcsundef	168
\@gls@link: added \@gls@saveentrycounter		\@codepage: replaced	
.....	82	\@ifundefined with	
added \@gls@setsort	82	\ifcsundef	21
\@gls@saveentrycounter: new	82	\gls@defglossaryentry: added	
\@gls@setupsort@def: new ...	11	\@gls@defsort	67
\@gls@setupsort@standard:		added short and long keys	63
new	10	replaced \@ifundefined with	
\@gls@setupsort@use: new ...	11	\ifcsundef	64
\@gls@xdy@locationlist: new	38	\gls@doclearpage: replaced	
\@glslink: replaced \@ifundefined		\@ifundefined with	
with \ifcsundef	91	\ifcsundef	34
\@glsnextpages: new	160	\glsadd: added \@gls@saveentrycounter	
\@makeglossary: Added check		138
for savewrites	146	\GlsAddXdyCounters: new	35
\@set@glo@numformat: added		\glentrycounterlabel: new	162
4th argument	84	\glentryitem: new	162
\@wrglossary: modified to take		\glentrylong: new	135
into account savewrites	150	\glentrylong: new	135
\@xdyattributelist: new	35	\glentrylongpl: new	136
General: added prefix to hyperlink		\glentrylongpl: new	135
.....	170	\glentryshort: new	135
etoolbox now loaded	4	\glentryshort: new	135
replaced \@ifundefined with		\glentryshortpl: new	135
\ifcsundef	24, 80, 159	\glentryshortpl: new	135
\acrfootnote: new	178	\glsgetgrouptitle: re-	
\ACRfull: added starred version	174	placed \@ifundefined with	
\Acrfull: added starred version	174	\ifcsundef	166
\acrfull: added starred version	173	\glsglossarymark: replaced	
\ACRfullpl: added starred ver-		\@ifundefined with	
sion	175	\ifcsundef	32
\Acrfullpl: added starred ver-		\glshyperlink: changed de-	
sion	175	fault from \glentryname to	
\acrfullpl: added starred ver-		\glentrytext	137
sion	175	\glshypernumber: replaced	
\acrlinkfootnote: new	178	\@ifundefined with	
\acrnolinkfootnote: new ...	178	\ifcsundef	169
\addglossarytocaptions: re-		\glsnumberformat: replaced	
placed \@ifundefined with		\@ifundefined with	
\ifcsundef	26	\ifcsundef	30
savewrites: new	22	\glsrefentry: new	161

<code>\glsresetsubentrycounter:</code>	<code>added \glsnextpages</code> 156
<code>new</code> 161	<code>make toctitle default to title</code> .. 156
<code>\glsseeitem:</code> <code>hyperlink</code> <code>uses</code>	<code>replaced \@ifundefined with</code>
<code>\glsseeitemformat</code> <code>instead</code>	<code>\ifcsundef</code> 155, 157
<code>of \glsentryname</code> 155	<code>\SetDescriptionFootnoteAcronymDisplayStyle:</code>
<code>\glsseeitemformat:new</code> 155	<code>expanded options link op-</code>
<code>\glssortnumberfmt:new</code> 10	<code>tions</code> 178
<code>\glsstepentry:new</code> 161	<code>\setentrycounter:</code> <code>added op-</code>
<code>\glsstepsubentry:new</code> 161	<code>tional argument</code> 167
<code>\glssubentrycounterlabel:</code>	<code>\showacronymlists:new</code> 199
<code>new</code> 162	<code>\showglocounter:new</code> 196
<code>\glssubentryitem:new</code> 162	<code>\showglodesc:new</code> 198
<code>theglossary:replaced \@ifundefined</code>	<code>\showglodescplural:new</code> ... 198
<code>with \ifcsundef</code> 162	<code>\showglofirst:new</code> 196
<code>short:new</code> 53	<code>\showglofirstpl:new</code> 196
<code>shortplural:new</code> 53	<code>\showgloflag:new</code> 199
<code>\ifglossaryexists:</code> <code>re-</code>	<code>\showgloindex:new</code> 199
<code>placed \@ifundefined with</code>	<code>\showglolevel:new</code> 195
<code>\ifcsundef</code> 44	<code>\showglongame:new</code> 197
<code>\ifglentryexists:</code> <code>re-</code>	<code>\showgloparent:new</code> 195
<code>placed \@ifundefined with</code>	<code>\showgloplural:new</code> 196
<code>\ifcsundef</code> 45	<code>\showglosort:new</code> 198
<code>\istfile:deprecated</code> 148	<code>\showglossaries:new</code> 199
<code>glossaryentry:new</code> 160	<code>\showglossarycounter:new</code> . 200
<code>glossarysubentry:new</code> 160	<code>\showglossaryentries:new</code> . 200
<code>\newglossary:added \gls@defsortcount</code>	<code>\showglossaryin:new</code> 199
. 49	<code>\showglossaryout:new</code> 199
<code>replaced \@ifundefined with</code>	<code>\showglossarytitle:new</code> ... 200
<code>\ifcsundef</code> 48	<code>\showglosymbol:new</code> 198
<code>\newglossaryentry:</code> <code>replaced</code>	<code>\showglosymbolplural:new</code> . 198
<code>\DeclareRobustCommand</code>	<code>\showglotext:new</code> 196
<code>with \newrobustcmd</code> 56	<code>\showglotype:new</code> 196
<code>\newglossarystyle:</code> <code>re-</code>	<code>\showglouserii:new</code> 197
<code>placed \@ifundefined with</code>	<code>\showglouseriii:new</code> 197
<code>\ifcsundef</code> 168	<code>\showglouseriv:new</code> 197
<code>entrycounter:new</code> 9	<code>\showglouserv:new</code> 197
<code>entrycounterwithin:new</code> 9	<code>\showglouservi:new</code> 197
<code>\oldacronym:replaced \@ifundefined</code>	<code>subentrycounter:new</code> 9
<code>with \ifcsundef</code> 172	<code>\writeist:</code> <code>added xindy-only</code>
<code>compatible-2.07: compatible-</code>	<code>macro definitions to glossary</code>
<code>2.07 option added</code> 22	<code>open tag</code> 142
<code>long:new</code> 53	<code>modified to support new for-</code>
<code>longplural:new</code> 54	<code>mat</code> 140
<code>nonumberlist: now boolean</code> ... 52	<code>\@glswritefiles:</code> <code>added check</code>
<code>sort:new</code> 9	<code>for empty glossaries</code> 148
<code>counter:replaced \@ifundefined</code>	<code>General: made robust</code> 95
<code>with \ifcsundef</code> 52	<code>\ACRfull: made robust</code> 174
<code>\printglossary:</code> <code>added</code>	
<code>\currentglossary</code> 156	

\Acrfull: made robust	174	\glssymbolplural: made robust	113
\acrfull: made robust	173	
\acrfullformat:	removed	\Glstext: made robust	103
\acronymfont as it should al-		\glstext: made robust	102
ready be set in the second ar-		\GLSuseri: made robust	116
gument.	174	\Glsuseri: made robust	115
\ACRfullpl: made robust	175	\glsuseri: made robust	115
\Acrfullpl: made robust	175	\GLSuserii: made robust	117
\acrfullpl: made robust	175	\Glsuserii: made robust	117
\ACRlong: made robust	129	\glsuserii: made robust	116
\Acrlong: made robust	128	\GLSuseriii: made robust	118
\acrlong: made robust	127	\Glsuseriii: made robust	118
\ACRlongpl: made robust	131	\glsuseriii: made robust	118
\Acrlongpl: made robust	130	\GLSuseriv: made robust	120
\acrlongpl: made robust	129	\Glsuseriv: made robust	119
\ACRshort: made robust	125	\glsuseriv: made robust	119
\Acrshort: made robust	124	\GLSuserv: made robust	121
\acrshort: made robust	123	\Glsuserv: made robust	121
\ACRshortpl: made robust	127	\glsuserv: made robust	120
\Acrshortpl: made robust	126	\GLSuservi: made robust	123
\acrshortpl: made robust	125	\Glsuservi: made robust	122
\Gls: made robust	94	\glsuservi: made robust	122
\glsadd: made robust	138		
\glsaddall: made robust	138	3.02	
\GLSdesc: made robust	110	\@@do@wrglossary: changed	
\Glsdesc: made robust	109	\@glslocref to \theglsentrycounter	
\glsdesc: made robust	109	152
\GLSdescplural: made robust	111	\@do@wrglossary: changed	
\Glsdescplural: made robust	111	\@do@wr@glossary to test for	
\glsdescplural: made robust	110	indexonlyfirst option; put old	
\glsfirst: made robust	103	\@do@wr@glossary code into	
\GLSfirstplural: made robust	107	\@@do@wrglossary	150
\Glsfirstplural: made robust	106	\@gls@missingnumberlist:	
\glsfirstplural: made robust	106	new	54
\glslink: made robust	81	\@glswritefiles: added check	
\GLSname: made robust	108	for existence of token in case	
\Glsname: made robust	108	\makeglossaries has been	
\glsname: made robust	107	omitted	148
\GLSpl: made robust	99	\@wrglossary: added check for	
\Glspl: made robust	98	glossary file defined	150
\glspl: made robust	96	General: added check for polyglos-	
\GLSplural: made robust	105	sia	24
\GLSsymbol: made robust	113	reversed order of package check	28
\Glsymbol: made robust	112	savenumberlist: new	7
\glssymbol: made robust	112	ucmark: new	9
\GLSsymbolplural: made robust		\gls@defglossaryentry: added	
.....	114	numberlist element	66
\Glsymbolplural: made robust		\gls@save@numberlist: new	155
.....	114	\glsdisplaynumberlist: new	136

<code>\glsentrycounter</code> : set default value	82	<code>index</code> : added check for <code>glsnogroupskip</code>	246
<code>\Glsentryfull</code> : fixed bug (replaced <code>\glsentryshortpl</code> with <code>\glsentryshort</code>)	136	<code>nogroupskip</code> : new	9
<code>\glsentryfullpl</code> : fixed bug (replaced <code>\glsentryshort</code> with <code>\glsentryshortpl</code>)	136	<code>long</code> : added check for <code>glsnogroupskip</code>	218
<code>\glsentrynumberlist</code> : new ..	136	<code>long3col</code> : added check for <code>glsnogroupskip</code>	220
<code>\glsmoveentry</code> : new	68	<code>long4col</code> : added check for <code>glsnogroupskip</code>	221
<code>\glsnumlistlastsep</code> : new ...	137	<code>longragged</code> : added check for <code>glsnogroupskip</code>	224
<code>\glsnumlistsep</code> : new	137	<code>longragged3col</code> : added check for <code>glsnogroupskip</code>	226
<code>\glsresetsubentrycounter</code> : new	161	<code>nopostdot</code> : new	9
<code>\ifglshaschildren</code> : new	45	<code>\printglossary</code> : allow title to override default <code>toctitle</code>	156
<code>\ifglshasparent</code> : new	46	<code>tree</code> : added check for <code>glsnogroupskip</code>	248
<code>\makeglossaries</code> : added list parser	148	<code>treenoname</code> : added check for <code>glsnogroupskip</code>	249
<code>indexonlyfirst</code> : new	20	<code>super</code> : added check for <code>glsnogroupskip</code>	234
<code>\printglossary</code> : add a way to fetch current entry label ...	156	<code>super3col</code> : added check for <code>glsnogroupskip</code>	235
<code>\renewglossarystyle</code> : new ..	168	<code>super4col</code> : added check for <code>glsnogroupskip</code>	237
<code>\showglossaryentries</code> : fixed misspelt command	200	<code>superragged</code> : added check for <code>glsnogroupskip</code>	240
<code>\SmallNewAcronymDef</code> : fixed broken short and long plural	187	<code>superragged3col</code> : added check for <code>glsnogroupskip</code>	242
3.03		3.04	
<code>\@gls@sanitizesort</code> : new	17	<code>\@do@wrglossary</code> : changed <code>\theglsentrycounter</code> back to <code>\@gls@locref</code>	152
<code>\@gls@setupsort@standard</code> : used <code>\@gls@sanitizesort</code> ..	10	modified to compensate for possible incorrect page number	151
General: allow title to set <code>toctitle</code>	158	<code>\@gls@escbsdq</code> : unsanitize <code>\gls@numberpage</code> , <code>\gls@alphpage</code> , <code>\gls@Alphpage</code> and <code>\gls@romanpage</code>	85
<code>\glsinlinedescformat</code> : new ..	214	General: Added check for doc package	4
<code>\glsinlineemptydescformat</code> : new	214	added <code>datatool-base</code> as a required package	4
<code>\glsinlinenameformat</code> : new ..	214	added local key	81
<code>\glsinlinepostchild</code> : new ..	213	<code>\gls@Alphpage</code> : new	150
<code>\glsinlinesubdescformat</code> : new	214	<code>\gls@alphpage</code> : new	150
<code>\glsinlinesubnameformat</code> : new	214		
<code>\glspostinline</code> : replaced “.” with <code>\glspostdescription</code>	214		
<code>altlongragged4col</code> : added check for <code>glsnogroupskip</code> ..	227		
<code>altsuperragged4col</code> : added check for <code>glsnogroupskip</code> ..	244		
<code>alttree</code> : added check for <code>glsnogroupskip</code>	252		

\gls@disablepagerefexpansion: new 150	\glossarypreamble: modified to work with \setglossarypreamble 31
\gls@numberpage: new 151	\gls@docclearpage: added check for openright 34
\gls@protected@pagefmts: new 150	\glspostdescription: Added spacefactor code 8
\gls@romanpage: new 151	\GlsSetXdyCodePage: Added check for fontspec 43
\glsdefmain: added check for doc package 12	\SetDescriptionAcronymDisplayStyle: now using \glsdoparenifnotempty 182
\glsorg@endtheglossary: new . 5	\setglossarypreamble: new .. 31
\glsorg@glossary: new 4	3.08a
\glsorg@theglossary: new 5	\@glo@storeentry: no longer need to check for special char- acters in any of the fields other than sort 69
\glsorg@wrglossary: new 4	updated for \glossentry 69
altlist: replaced \newline with paragraph break 216	\@glossaryentryfield: switched to \glossentry 68
\PrintChanges: new 5	\@glossarysubentryfield: switched to \subglossentry 68
\printglossary: Moved aux write to end of document to prevent unwanted whatsit oc- curring here. 157	General: added nogroupskip key to \printglossary 159
3.05	removed definition of \@glossaryentryfield .. 288
\@do@wrglossary: add Roman case. Fixed bugs in the else statements 151	removed definition of \@glossarysubentryfield 288
\@gls@link: added check for “no- hypertypes” 82	\compatibleglossentry: new 163
\@gls@nohyperlist: new 15	\compatiblesubglossentry: new 164
mcolalttree: replaced ‘2’ with \glsmcols 232	\glossaryentryfield: depre- cated 165
mcolindex: replaced ‘2’ with \glsmcols 229	\Glossentrydesc: new 164
mcoltree: replaced ‘2’ with \glsmcols 230	\glossentrydesc: new 164
mcoltreenoname: replaced ‘2’ with \glsmcols 231	\Glossentryname: new 163
\gls@protected@pagefmts: added Roman to list 150	\glossentryname: new 163
\gls@Romanpage: new 151	\Glossentrysymbol: new 164
\GlsDeclareNoHyperList: new 15	\glossentrysymbol: new 164
\glsgetgrouplabel: fixed bug (typo in \equal) 167	\gls@assign@desc@field: new 17
\nopostdesc: made robust 28	\gls@assign@descplural@field: new 17
nohypertypes: new 15	\gls@assign@field: new 56
3.06	\gls@ifnotmeasuring: new ... 70
\@xdy@main@language: Changed back to using \language name 21	\glsaddallunused: new 139
\findrootlanguage: Obsoleted 42	\glsexpandfields: new 56
3.07	\glsnoexpandfields: new 56
\@gls@link: fixed bug that failed to find entry in list 82	\glssee: made robust 154

<code>\glseeformat</code> : made robust .. 154	<code>\Glsentrydescplural</code> : made robust .. 132
<code>\glseeitem</code> : made robust 155	<code>\Glsentryfirst</code> : made robust . 133
<code>\glseeelist</code> : made robust 154	<code>\Glsentryfirstplural</code> : made robust .. 133
<code>\ifglshdescsuppressed</code> : new .. 46	<code>\Glsentryfull</code> : made robust .. 136
<code>\ifglshasdesc</code> : new 46	<code>\Glsentryfullpl</code> : made robust 136
<code>\ifglshassymbol</code> : new 46	<code>\Glsentrylong</code> : made robust .. 135
<code>list</code> : updated list style to use <code>\glossentry</code> and <code>\subglossentry</code> 214	<code>\Glsentrylongpl</code> : made robust 136
<code>listdotted</code> : updated listdotted style to use <code>\glossentry</code> and <code>\subglossentry</code> 217	<code>\Glsentryname</code> : made robust .. 131
<code>altlist</code> : updated altlist style to use <code>\glossentry</code> and <code>\subglossentry</code> 215	<code>\Glsentryplural</code> : made robust 132
<code>atlongragged4col</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code> 227	<code>\Glsentryshort</code> : made robust . 135
<code>alttree</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code> 250	<code>\Glsentryshortpl</code> : made robust 135
<code>index</code> : added paragraph break at end of environment 245	<code>\Glsentrysymbol</code> : made robust 133
updated to use <code>\glossentry</code> and <code>\subglossentry</code> 246	<code>\Glsentrysymbolplural</code> : made robust 133
<code>inline</code> : updated inline style to use <code>\glossentry</code> and <code>\subglossentry</code> 212	<code>\Glsentrytext</code> : made robust .. 132
<code>long</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code> 218	<code>\Glsentryuseri</code> : made robust . 134
<code>longragged</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code> 224	<code>\Glsentryuserii</code> : made robust 134
<code>longragged3col</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code> 225	<code>\Glsentryuseriii</code> : made robust 134
<code>tree</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code> 247	<code>\Glsentryuseriv</code> : made robust 134
<code>\setglossarystyle</code> : new 167	<code>\Glsentryuserv</code> : made robust . 135
<code>\setglossentrycompatibility</code> : new 165	<code>\Glsentryuservi</code> : made robust 135
<code>superragged</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code> 240	<code>\glstextup</code> : new 173
3.09a	<code>\if@gl@docloaded</code> : Add a fix for <code>\RecordChanges</code> 4
<code>\@gl@assign@symbolplural@field</code> : new 17	<code>\ifglshassymbol</code> : changed test to check for <code>\@gl@default@symbol</code> 46
<code>\@gl@default@value</code> : new ... 51	3.10a
<code>\Glsentrydesc</code> : made robust .. 132	<code>\@gl@keymap</code> : new 58
	<code>\@gl@provide@newglossary</code> : new 47
	<code>\@gl@defaultplural</code> : Obsolete . 54
	<code>\@gl@nodelsc</code> : new 54
	<code>\gl@assign@type@field</code> : new 17
	<code>\gl@defglossaryentry</code> : Changed to using <code>\@gl@default@value</code> 63
	new 62
	<code>\glswritedefhook</code> : new 61
	<code>\makeglossaries</code> : Added providecommand code to aux file 147
	<code>\new@glossaryentry</code> : new 57
	<code>\newglossary</code> : added <code>\@gl@provide@newglossary</code> 48

\printglossary: Added provide-	removed \makefirstuc (now
command code to aux file 157, 158	dealt with in \glsentryfmt) 99
3.11a	\@acrlong: added \glslabel,
\@ACRlong: added \glslabel,	\glsifplural, \glscapscase,
\glsifplural, \glscapscase,	\glsinsert and \glscustomtext
\glsinsert and \glscustomtext 287
..... 288	\@acrshort: added \glslabel,
\@ACRshort: added \glslabel,	\glsifplural, \glscapscase,
\glsifplural, \glscapscase,	\glsinsert and \glscustomtext
\glsinsert and \glscustomtext 286
..... 287	\@gls@: add \glslabel,
\@Acrlong: added \glslabel,	\glsifplural, \glscapscase,
\glsifplural, \glscapscase,	\glscustomtext and
\glsinsert and \glscustomtext	\glsinsert 93
..... 287	change to using \glsentryfmt
\@Acrshort: added \glslabel,	style commands 93
\glsifplural, \glscapscase,	\@gls@noexpand@fields: Fixed
\glsinsert and \glscustomtext	bug expand replaced with
..... 286	noexpand 55
\@GLS@: add \glslabel,	\@glsdisp: add \glslabel,
\glsifplural, \glscapscase,	\glsifplural, \glscapscase,
\glscustomtext and	\glscustomtext and
\glsinsert 96	\glsinsert 101
change to using \glsentryfmt	change to using \glsentryfmt
style commands 96	style commands 101
removed \MakeUppercase	\@glspl@: add \glslabel,
(now moved to \glsentryfmt)	\glsifplural, \glscapscase,
..... 96	\glscustomtext and
\@GLSpl: add \glslabel,	\glsinsert 97
\glsifplural, \glscapscase,	General: added \glslabel,
\glscustomtext and	\glsifplural, \glscapscase,
\glsinsert 100	\glsinsert and \glscustomtext
change to using \glsentryfmt 124–131
style commands 100	changed to just use \Glsentrydescplural
removed \MakeUppercase 111
as now dealt with in	changed to just use \glsentrydescplural
\glsentryfmt 100 111, 112
\@Gls@: add \glsifplural,	changed to just use \Glsentrydesc
\glscapscase, \glscustomtext 110
and \glsinsert 94	changed to just use \glsentrydesc
change to using \glsentryfmt 109, 110
style commands 94	changed to just use \Glsentryfirstplural
removed \makefirstuc (now 107
dealt with in \glsentryfmt) 95	changed to just use \glsentryfirstplural
\@Glspl@: add \glsifplural, 106, 107
\glscapscase, \glscustomtext	changed to just use \Glsentryfirst
and \glsinsert 98 104
change to using \glsentryfmt	changed to just use \glsentryfirst
style commands 98 103, 104

changed to just use <code>\Glsentryname</code>	<code>\defglstdisplayfirst:</code> obso-
..... 108	leted 79
changed to just use <code>\glentryname</code>	<code>\defglentryfmt:</code> new 47
..... 108, 109	<code>\forglentries:</code> replaced <code>\ifx</code>
changed to just use <code>\Glsentryplural</code>	with <code>\ifdefempty</code> 44
..... 105	<code>\gls@assign@desc:</code> new 61
changed to just use <code>\glentryplural</code>	<code>\gls@defglossaryentry:</code> Fixed
..... 105, 106	default counter if none sup-
changed to just use <code>\Glsentrysymbolplural</code>	plied 66
..... 114	<code>\gls@doentryfmt:</code> new 47
changed to just use <code>\glentrysymbolplural</code>	<code>\glstdisplay:</code> obsoleted 79
..... 114, 115	<code>\glstdisplayfirst:</code> obsoleted .. 79
changed to just use <code>\Glsentrysymbol</code>	<code>\glsgenentryfmt:</code> new 76
..... 113	<code>\glsgetgrouptitle:</code> Added
changed to just use <code>\glentrysymbol</code>	check in case non-Latin alpha-
..... 112, 113	bet in use 166
Changed to just use	<code>\glsglossarymark:</code> replaced
<code>\Glsentrytext</code> 103	<code>\MakeUppercase</code> with
changed to just use <code>\glentrytext</code>	<code>\mfirstucMakeUppercase</code> . 32
..... 102	<code>\glsnavigation:</code> switched to us-
changed to just use <code>\Glsentryuseriii</code>	ing <code>\@gls@getgrouptitle</code> 211
..... 118	<code>\ifglshasdesc:</code> replaced
changed to just use <code>\glentryuseriii</code>	<code>\ifdefempty</code> with <code>\ifcsemtyp</code>
..... 118, 119 46
changed to just use <code>\Glsentryuserii</code>	<code>\ifglshaslong:</code> new 46
..... 117	<code>\ifglshasshort:</code> new 47
changed to just use <code>\glentryuserii</code>	<code>\ifglshassymbol:</code> replaced
..... 116, 117	<code>\ifdefempty</code> with <code>\ifcsemtyp</code>
changed to just use <code>\Glsentryuseriv</code> 46
..... 120	<code>\ifglused:</code> replaced <code>\ifthenelse</code>
changed to just use <code>\glentryuseriv</code>	with <code>\ifbool</code> 45
..... 120	<code>\longnewglossaryentry:</code> new . 62
changed to just use <code>\Glsentryuseri</code>	<code>\newglossary:</code> replaced
..... 116	<code>\glstdisplay</code> and <code>\glstdisplayfirst</code>
changed to just use <code>\glentryuseri</code>	with <code>\glentryfmt</code> 48
..... 115, 116	compatible-3.07: cnew 22
changed to just use <code>\Glsentryuservi</code>	<code>\SetCustomDisplayStyle:</code> up-
..... 123	dated to use <code>\defglentryfmt</code>
changed to just use <code>\glentryuservi</code> 192
..... 122, 123	<code>\SetDefaultAcronymDisplayStyle:</code>
changed to just use <code>\Glsentryuserv</code>	changed to use <code>\defglentryfmt</code>
..... 121 176
changed to just use <code>\glentryuserv</code>	<code>\SetDescriptionAcronymDisplayStyle:</code>
..... 121, 122	updated to use <code>\defglentryfmt</code>
Now requires <code>textcase</code> 3 182
<code>acronymlists:</code> replaced	<code>\SetDescriptionDUAAcronymDisplayStyle:</code>
<code>\@addtoacronymlists</code> with	updated to use <code>\defglentryfmt</code>
<code>\DeclareAcronymList</code> 15 180
<code>\defglstdisplay:</code> obsoleted 79	

\SetDescriptionFootnoteAcronymDisplayStyle:	\gls@assign@name@field:
updated to use \defglsentryfmt	changed to use \glssetnoexpandfield
..... 178 17
\SetDUADisplayStyle: updated	\gls@assign@type@field:
to use \defglsentryfmt .. 189	changed to use \glssetexpandfield
\SetFootnoteAcronymDisplayStyle: 17
updated to use \defglsentryfmt	\gls@checkseeallowed: new .. 52
..... 184	\glsaddallunused: set default to
\SetSmallAcronymDisplayStyle:	\@glo@types 139
updated to use \defglsentryfmt	\Glsentryfull: changed to use
..... 187	\acrfullformat 136
\setupglossaries: new 23	\glsentryfull: changed to use
\showglo long: new 198	\acrfullformat 136
\showglo short: new 198	\Glsentryfullpl: changed to
numbers: new 23	use \acrfullformat 136
symbols: new 22	\glsentryfullpl: changed to
3.12a	use \acrfullformat 136
\gls@defglossaryentry: added	\gls glossarymark: renamed
\glslabel 62	\glossarymark to \gls glossarymark
\glsaddkey: new 59	to avoid conflict with memoir 32
3.13a	\glsprestandardsort: new ... 10
\@gls@assign@symbol@field:	\glssetnoexpandfield: new .. 16
changed to use \glssetnoexpandfield	\alt super 4 col header: switched
..... 17	to \tabularnewline 238
\@gls@assign@symbolplural@field:	\alt super 4 col header border:
changed to use \glssetnoexpandfield	switched to \tabularnewline
..... 17 239
\@gls@link: removed \relax .. 82	long: switched to \tabularnewline
\@gls@notranslatorhook: new 19 218
\@gls@setupsort@standard:	long 3 col: switched to \tabularnewline
moved \@gls@santizesort 219
to \glsprestandardsort .. 10	long 3 col header: switched to
General: added cs@gls@notranslatorhook	\tabularnewline 220
to else clause 28	long 3 col header border: switched
ucmark: added check for memoir . 9	to \tabularnewline 220
see: added \gls@checkseeallowed	long 4 col: switched to \tabularnewline
..... 52 221
\glossarysection: changed	long 4 col header: switched to
\glossarymark to \gls glossarymark	\tabularnewline 221
..... 32	long header: switched to
\glossarystyle: fixed bug	\tabularnewline 219
caused by using \ifdef in-	long header border: switched to
stead of \ifcsdef 168	\tabularnewline 219
\gls@assign@desc@field:	\SetFootnoteAcronymDisplayStyle:
changed to use \glssetnoexpandfield	fixed missing argument bug 185
..... 17	super: switched to \tabularnewline
\gls@assign@descplural@field: 233
changed to use \glssetnoexpandfield	super 3 col: switched to
..... 17	\tabularnewline 235

super3colheader: switched to \tabularnewline 236	check for existence of default glossary 64
super4col: switched to \tabularnewline 237	set the default for firstplural to be the value of plural 65
super4colheader: switched to \tabularnewline 237	xindygloss:new 22
super4colheaderborder: switched to \tabularnewline 238	\longprovideglossaryentry: new 62
superheader: switched to \tabularnewline 234	compatible-2.07: added check for 2.07 before setting 3.07 compatibility 22
superheaderborder: switched to \tabularnewline 234	nottranslate:new 19
3.14a	\provideglossaryentry:new . 57
\@glswritefiles: renamed \glswritefiles to \@glswritefiles and used “savewrites” option to set \glswritefiles 148	\gls@defglossaryentry: added check for first key 65
General:new 201	General: fixed non-value options so that they can be passed to document class 7
acronyms:new 13	\CustomAcronymFields: in- serted missing comma 192
\gls@defglossaryentry: added	

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	151
\@glossarysec	5
\@glossaryseclabel	6
\@glossarysecstar	6
\@gls@default@entryfmt	282
\@gls@expand@field	55
\@ACRlong	288
\@ACRshort	286
\@Acrlong	287
\@Acrshort	286
\@GLS@	95
\@GLSpl	99
\@Gls@	94
\@Glspl@	98
\@PGLS	206
\@PGLS@	206
\@PGLSpl	207
\@PGLSpl@	207
\@PglS	204
\@PglS@	204
\@PglSpl	205
\@PglSpl@	205
\@acrlong	287
\@acrshort	286
\@addtoacronymlists	14
\@delimN	170
\@delimR	170
\@disable@onlypremakeg	25
\@disable@premakecs	25
\@disabled@glSaddxdycounters	36
\@do@seeglossary	153
\@do@wrglossary	150, 254
\@glo@seeautonumberlist	7
\@glo@storeentry	68
\@glo@types	47
\@glossary	149
\@glossary@default@style	6
\@glossaryentryfield	68
\@glossarysection	33
\@glossarysubentryfield	68
\@gls	93
\@gls@	93
\@gls@@link	81
\@gls@addpredefinedattributes	37
\@gls@assign@symbol@field ...	17
\@gls@assign@symbolplural@field	17
\@gls@checkactual	89
\@gls@checkboxar	88
\@gls@checkescactual	87
\@gls@checkescbar	87
\@gls@checkesclevel	88
\@gls@checkescquote	86
\@gls@checklevel	89
\@gls@checkmkidxchars	85
\@gls@checkquote	86
\@gls@codepage	42
\@gls@counterwithin	9
\@gls@declareoption	7
\@gls@default@value	51
\@gls@do@acronymsdef	13
\@gls@escbsdq	84
\@gls@expand@fields	55
\@gls@fixbraces	153
\@gls@getcounter	49
\@gls@getcounterprefix	152
\@gls@getgrouptitle	166
\@gls@hypergroup	211
\@gls@ifinlist	35
\@gls@keymap	58, 201
\@gls@link	82
\@gls@loadlist	8
\@gls@loadlong	7
\@gls@loadsuper	8
\@gls@loadtree	8
\@gls@makefirstuc	208
\@gls@missingnumberlist	54
\@gls@noaccess	275
\@gls@noexpand@field	55
\@gls@noexpand@fields	55
\@gls@nohyperlist	15
\@gls@notranslatorhook	19
\@gls@onlypremakeg	25
\@gls@provide@newglossary ...	47

<code>\@gls@renewglossary</code>	149	<code>\@glswidestname</code>	250
<code>\@gls@sanitizedesc</code>	16	<code>\@glswritefiles</code>	148
<code>\@gls@sanitizename</code>	17	<code>\@istfilename</code>	29
<code>\@gls@sanitizesort</code>	17	<code>\@makeglossary</code>	146
<code>\@gls@sanitizesymbol</code>	17	<code>\@newglossary</code>	49
<code>\@gls@saveentrycounter</code>	82	<code>\@newglossaryentryposthook</code>	67
<code>\@gls@setacrstyle</code>	20	<code>\@newglossaryentryprehook</code>	67
<code>\@gls@setcounter</code>	49	<code>\@no@post@desc</code>	28
<code>\@gls@setupsort@def</code>	11	<code>\@nopostdesc</code>	28
<code>\@gls@setupsort@standard</code>	10	<code>\@onlypremakeg</code>	25
<code>\@gls@setupsort@use</code>	11	<code>\@p@glossarysection</code>	33
<code>\@gls@startswithexpandonce</code>	56	<code>\@pgls</code>	203
<code>\@gls@tmpb</code>	86	<code>\@pgls@</code>	203
<code>\@gls@toc</code>	34	<code>\@pglspl</code>	204
<code>\@gls@updatechecked</code>	86	<code>\@pglspl@</code>	204
<code>\@gls@xdy@Lclass@Alpha-page-numbers</code>	39	<code>\@sPGLS</code>	206
<code>\@gls@xdy@Lclass@Appendix-page-numbers</code>	39	<code>\@sPGLSpl</code>	206
<code>\@gls@xdy@Lclass@Roman-page-numbers</code>	39	<code>\@sPgls</code>	204
<code>\@gls@xdy@Lclass@alpha-page-numbers</code>	39	<code>\@sPglspl</code>	205
<code>\@gls@xdy@Lclass@arabic-page-numbers</code>	39	<code>\@set@glo@numformat</code>	84, 255
<code>\@gls@xdy@Lclass@arabic-section-numbers</code>	39	<code>\@sgls</code>	93, 101
<code>\@gls@xdy@Lclass@roman-page-numbers</code>	38	<code>\@sgls@link</code>	81
<code>\@gls@xdy@locationlist</code>	38	<code>\@spgls</code>	203
<code>\@gls@xdy@checkbackslash</code>	90	<code>\@spglspl</code>	204
<code>\@gls@xdy@checkquote</code>	90	<code>\@wrglossary</code>	150
<code>\@gls@Alpha compositor</code>	29, 39	<code>\@xdy@main@language</code>	21
<code>\@gls@sacronymlists</code>	13	<code>\@xdy@attributelist</code>	35
<code>\@gls@defaultplural</code>	54	<code>\@xdy@attributes</code>	35
<code>\@gls@defaultsort</code>	54	<code>\@xdy@language</code>	42
<code>\@gls@disp</code>	101	<code>\@xdy@lettergroups</code>	43
<code>\@gls@firstletter</code>	139	<code>\@xdy@locationclassorder</code>	41
<code>\@gls@hypernumber</code>	169	<code>\@xdy@locref</code>	35
<code>\@gls@link</code>	91	<code>\@xdy@requiredstyles</code>	41
<code>\@gls@minrange</code>	140	<code>\@xdy@sortrules</code>	41
<code>\@gls@nextpages</code>	160	<code>\@xdy@useralphabets</code>	38
<code>\@gls@nodesc</code>	54	<code>\@xdy@userlocationdefs</code>	40
<code>\@gls@noname</code>	54	<code>\@xdy@userlocationnames</code>	40
<code>\@gls@nonextpages</code>	159		
<code>\@gls@openfile</code>	146		
<code>\@gls@order</code>	21		
<code>\@glspl@</code>	97		
<code>\@gls@target</code>	91		
		A	
		<code>\Ac</code>	194
		<code>\ac</code>	194
		<code>access (key)</code>	274
		<code>accsupp package</code>	273
		<code>\accsuppglossaryentryfield</code>	289
		<code>\accsuppglossarysubentryfield</code>	289
		<code>\Acf</code>	194
		<code>\acf</code>	194
		<code>\Acfp</code>	194

`\defglssdisplay` 79
`\defglssdisplayfirst` 79
`\defglssentry` 48
`\defglssentryfmt` 47, 50, 51, 73
`\DefineAcronymSynonyms` 193
`\delimN` 31, 169
`\delimR` 31, 169
`description` (environment) 214
`description` (key) 50
`description` (option) 20
`descriptionaccess` (key) 274
`\DescriptionDUANewAcronymDef` 180
`\DescriptionFootnoteNewAcronymDef`
..... 179, 290
`\descriptionname` 26
`\DescriptionNewAcronymDef` ..
..... 183, 291
`descriptionplural` (key) 50
`descriptionpluralaccess` (key) 275
`doc` package 4, 5, 12
`dua` (option) 20
`\DUANewAcronymDef` 189

E

`entrycounter` (option) 9
`entrycounterwithin` (option) 9
`\entryname` 26
environments:
 `align` 70, 82, 83
 `description` 214
 `longtable` 8, 195, 217–228
 `multicols` 229
 `supertabular` ... 8, 195, 232–245
 `theglossary` 5, 31, 162,
 163, 168, 230, 231, 247, 248, 250
 `theindex` 245
`equation` counter 82, 83
`etoolbox` package 4, 207

F

file types
 `.aux` 157
 `.glo` 68
 `.ist` 139, 145, 146
 `.toc` 34
 `.xdy` 29
 `glo` 200
`\findrootlanguage` 42
`first` (key) 51
`firstaccess` (key) 274

`\firstacronymfont` 176
`firstplural` (key) 51
`firstpluralaccess` (key) 274
`footnote` (option) 20
`\FootnoteNewAcronymDef` . 185, 291
`\foralllglossaries` 43
`\foralllglsentries` 44
`\forallrglsentries` 44

G

`garamondx` package 173
`\glolinkprefix` 82
`glossareentry` counter 161
`glossaries` package 42,
 43, 140, 195, 201, 214, 253, 273
`glossaries-accsupp` package ... 68, 273
`\GlossariesWarning` 16
`\GlossariesWarningNoLine` 16
`\glossary` 48, 146, 149, 167
`glossary` counters:
 `glossaryentry` 160
 `glossarysubentry` 160
`glossary` keys:
 `access` 274
 `counter` 52
 `description` 50
 `descriptionaccess` 274
 `descriptionplural` 50
 `descriptionpluralaccess` . 275
 `first` 51
 `firstaccess` 274
 `firstplural` 51
 `firstpluralaccess` 274
 `long` 53
 `longaccess` 275
 `longplural` 54
 `longpluralaccess` 275
 `name` 50
 `nonumberlist` 52
 `parent` 52
 `plural` 51
 `pluralaccess` 274
 `see` 52
 `short` 53
 `shortaccess` 275
 `shortplural` 53
 `shortpluralaccess` 275
 `sort` 50
 `symbol` 51

symbolaccess	274
symbolplural	51
symbolpluralaccess	274
text	51
textaccess	274
type	52
user1	53
user2	53
user3	53
user4	53
user5	53
user6	53
glossary package	1, 172
glossary styles:	
altlist	215, 216, 261
altlist	215
altlistgroup	216, 261
altlistgroup	216
altlisthypergroup ...	216, 261
altlisthypergroup	216
altlong4col ..	222, 223, 227, 263
altlong4col	222
altlong4colborder ...	223, 263
altlong4colborder	223
altlong4colheader ...	222, 263
altlong4colheader	222
altlong4colheaderborder .	
.....	223, 263
altlong4colheaderborder .	223
altlongragged4col	227, 228, 264
altlongragged4col	227
altlongragged4colborder .	
.....	228, 265
altlongragged4colborder .	228
altlongragged4colheader .	
.....	227, 265
altlongragged4colheader .	227
altlongragged4colheaderborder	
.....	228, 265
altlongragged4colheaderborder	
.....	228
altsuper4col .	238, 239, 243, 272
altsuper4col	238
altsuper4colborder ..	238, 272
altsuper4colborder	238
altsuper4colheader ..	238, 272
altsuper4colheader	238
altsuper4colheaderborder	
.....	239, 272
altsuper4colheaderborder	239
altsuperragged4col	243–245, 270
altsuperragged4col	243
altsuperragged4colborder	
.....	244, 271
altsuperragged4colborder	244
altsuperragged4colheader	
.....	244, 270
altsuperragged4colheader	244
altsuperragged4colheaderborder	
.....	245, 271
altsuperragged4colheaderborder	
.....	245
alttree	231, 250, 252, 267
alttree	250
alttreegroup	252, 268
alttreegroup	252
alttreehypergroup ...	252, 268
alttreehypergroup	252
index	229, 245–247, 265
index	245
indexgroup	246, 247, 265
indexgroup	246
indexhypergroup	247, 266
indexhypergroup	247
inline	260
inline	212
list	6, 214–216, 260
list	214
listdotted	216, 217, 261
listdotted	216
listgroup	215, 261
listgroup	215
listhypergroup	215, 261
listhypergroup	215
long	218, 219, 224, 262, 263
long	218
long3col	219, 220, 262
long3col	219
long3colborder	220, 262
long3colborder	220
long3colheader	220, 262
long3colheader	220
long3colheaderborder	220, 262
long3colheaderborder ...	220
long4col	220–222, 263
long4col	220
long4colborder	221, 263
long4colborder	221

long4colheader	221, 263	mcoltreenonamegroup	231
long4colheader	221	mcoltreenonamehypergroup	
long4colheaderborder	222, 263	231, 269
long4colheaderborder	222	mcoltreenonamehypergroup	231
longborder	218, 262	sublistdotted	261
longborder	218	sublistdotted	217
longheader	219, 262	super	233–235, 241, 271
longheader	219	super	233
longheaderborder	219, 262	super3col	235, 236, 271
longheaderborder	219	super3col	235
longragged	224, 225	super3colborder	235, 271
longragged	224	super3colborder	235
longragged3col ...	225, 226, 264	super3colheader	236, 272
longragged3col	225	super3colheader	236
longragged3colborder	226, 264	super3colheaderborder	236, 272
longragged3colborder	226	super3colheaderborder ...	236
longragged3colheader	226, 264	super4col	236–238, 272
longragged3colheader	226	super4col	236
longragged3colheaderborder		super4colborder	237, 272
.....	226, 264	super4colborder	237
longragged3colheaderborder		super4colheader	237, 272
.....	226	super4colheader	237
longraggedborder	224, 264	super4colheaderborder	238, 272
longraggedborder	224	super4colheaderborder ...	238
longraggedheader	225, 264	superborder	234, 271
longraggedheader	225	superborder	234
longraggedheaderborder	225, 264	superheader	234, 271
longraggedheaderborder ..	225	superheader	234
mcolalttree	232, 269	superheaderborder ...	234, 271
mcolalttree	231	superheaderborder	234
mcolalttreegroup	232, 269	superragged	240, 241, 269
mcolalttreegroup	232	superragged	240
mcolalttreehypergroup	232, 269	superragged3col ..	241–243, 270
mcolalttreehypergroup ...	232	superragged3col	241
mcolindex	229, 268	superragged3colborder	242, 270
mcolindex	229	superragged3colborder ...	242
mcolindexgroup	229, 268	superragged3colheader	242, 270
mcolindexgroup	229	superragged3colheader ...	242
mcolindexhypergroup .	229, 268	superragged3colheaderborder	
mcolindexhypergroup	229	243, 270
mcoltree	230, 268	superraggedborder ...	240, 270
mcoltree	230	superraggedborder	240
mcoltreegroup	269	superraggedheader ...	241, 270
mcoltreegroup	230	superraggedheader	241
mcoltreehypergroup ..	230, 269	superraggedheaderborder .	
mcoltreehypergroup	230	241, 270
mcoltreenoname	231, 269	superraggedheaderborder .	241
mcoltreenoname	231	superraggedright3colheaderborder	
mcoltreenonamegroup .	231, 269	243

tree	230, 247, 248, 250, 266	106, 107, 109, 110, 112, 113,
tree	247	115–117, 119, 120, 122, 161, 203
treegroup	230, 248, 266	\gls@Alphpage
treegroup	248	150
treehypergroup	248, 266	\gls@alphpage
treehypergroup	248	150
treenoname ...	231, 248, 249, 266	\gls@assign@desc
treenoname	248	61
treenonamegroup	249, 267	\gls@assign@desc@field
treenonamegroup	249	17
treenonamehypergroup	249, 267	\gls@assign@descplural@field
treenonamehypergroup	249	17
glossary-hypernav package	139	\gls@assign@field
glossary-list package	6, 8, 214	56
glossary-long package ..	8, 217, 227, 233	\gls@assign@name@field
glossary-longragged package	223	17
glossary-mcols package	229	\gls@assign@type@field
glossary-super package		17
.....	8, 217, 232, 239, 243	\gls@checkisacronymlist
glossary-superragged package	239	14
glossary-tree package	8, 245	\gls@checkseeallowed
glossaryentry (counter)	160	52
glossaryentry counter	9, 161, 162	\gls@codepage
\glossaryentryfield ..	165, 168, 169	21
\glossaryentrynumber	160	\gls@defglossaryentry
\glossaryentrynumbers		62
.....	7, 30, 156, 158	\gls@disablepagerefexpansion
\glossaryheader	163, 168	150
\glossarymark	32	\gls@doclearpage
\glossaryname	25, 26	34
\glossarypostamble	32, 168	\gls@doentryfmt
\glossarypreamble	31, 168	47
\glossarysection	6, 32, 48	\gls@hypergroup prerun
\glossarystyle	168, 195	210
glossarysubentry (counter) ...	160	\gls@ifnotmeasuring
glossarysubentry counter ...	9, 161, 162	70
\glossarysubentryfield	165	\gls@level
\glossentry	51, 163	54
\Glossentrydesc	164	\gls@numberpage
\glossentrydesc	164, 288	151
\Glossentryname	163	\gls@protected@pagefmts
\glossentryname	163, 288	150
\Glossentrysymbol	164	\gls@Romanpage
\glossentrysymbol	164, 288	151
\GLS	95	\gls@romanpage
\Gls	94, 98, 207	151
\gls	4,	\gls@save@numberlist
51, 72, 81, 93, 95, 96, 102–104,		155
		\gls@suffixF
		30
		\gls@suffixFF
		30
		\glsaccessdisplay
		281
		\glsacccsupp
		278
		\glsadd
		72, 137, 138, 167
		\glsadd options
		counter
		138
		format
		138, 169
		\glsaddall
		72, 138
		\glsaddall options
		types
		138
		\glsaddallunused
		138
		\glsaddkey
		59
		\GlsAddLetterGroup
		43
		\GlsAddSortRule
		41
		\GlsAddXdyAlphabet
		38
		\GlsAddXdyAttribute
		36, 253
		\GlsAddXdyCounters
		35, 253
		\GlsAddXdyLocation
		40, 254
		\GlsAddXdyStyle
		42
		\glsautoprefix
		6
		\glsclearpage
		34
		\glsclosebrace
		139
		\glscompositor
		29, 39

<code>\glscounter</code>	15, 49	<code>\glstentrylongpluralaccess</code> ..	278
<code>\GlsDeclareNoHyperList</code>	15	<code>\Glsentryname</code>	131
<code>\glsdefaulttype</code>	12	<code>\glstentryname</code>	131, 155
<code>\glsdefmain</code>	12	<code>\glstentrynumberlist</code>	136
<code>\GLSdesc</code>	110	<code>\Glsentryplural</code>	132
<code>\Glsdesc</code>	109	<code>\glstentryplural</code>	132
<code>\glsdesc</code>	109, 110	<code>\glstentrypluralaccess</code>	277
<code>\GLSdescplural</code>	111	<code>\Glsentryprefix</code>	202
<code>\Glsdescplural</code>	111	<code>\glstentryprefix</code>	202
<code>\glsdescplural</code>	110, 111	<code>\Glsentryprefixfirst</code>	202
<code>\glsdescriptionaccessdisplay</code>	280	<code>\glstentryprefixfirst</code>	202
<code>\glsdescriptionpluralaccessdisplay</code>	280	<code>\Glsentryprefixfirstplural</code> .	202
.....	280	<code>\glstentryprefixfirstplural</code> .	202
<code>\glsdescwidth</code> ...	217, 223, 233, 239	<code>\Glsentryprefixplural</code>	202
<code>\glsdisablehyper</code>	92	<code>\glstentryprefixplural</code>	202
<code>\glsdisp</code>	100	<code>\Glsentryshort</code>	135
<code>\glsdisplay</code>	72, 79, 92	<code>\glstentryshort</code>	135
<code>\glsdisplayfirst</code>	72, 79, 92	<code>\glstentryshortaccess</code>	278
<code>\glsdisplaynumberlist</code>	136	<code>\Glsentryshortpl</code>	135
<code>\glsdoifexists</code>	45	<code>\glstentryshortpl</code>	135
<code>\glsdoifnoexists</code>	45	<code>\glstentryshortpluralaccess</code> .	278
<code>\glsdoparenifnotempty</code>	186	<code>\glstentrysort</code>	134
<code>\glsenablehyper</code>	92	<code>\Glsentrysymbol</code>	133
<code>\glstentryaccess</code>	277	<code>\glstentrysymbol</code>	133
<code>\glstentrycounter</code>	82	<code>\glstentrysymbolaccess</code>	277
<code>\glstentrycounterlabel</code>	162	<code>\Glsentrysymbolplural</code>	133
<code>\Glsentrydesc</code>	132	<code>\glstentrysymbolplural</code>	133
<code>\glstentrydesc</code>	132	<code>\glstentrysymbolpluralaccess</code>	278
<code>\glstentrydescaccess</code>	278	<code>\Glsentrytext</code>	132
<code>\Glsentrydescplural</code>	132	<code>\glstentrytext</code>	132, 155
<code>\glstentrydescplural</code>	132	<code>\glstentrytextaccess</code>	277
<code>\glstentrydescpluralaccess</code> ..	278	<code>\glstentrytype</code>	134
<code>\Glsentryfirst</code>	133	<code>\Glsentryuseri</code>	134
<code>\glstentryfirst</code>	133	<code>\glstentryuseri</code>	134
<code>\glstentryfirstaccess</code>	277	<code>\Glsentryuserii</code>	134
<code>\Glsentryfirstplural</code>	133	<code>\glstentryuserii</code>	134
<code>\glstentryfirstplural</code>	133	<code>\Glsentryuseriii</code>	134
<code>\glstentryfirstpluralaccess</code> .	277	<code>\glstentryuseriii</code>	134
<code>\glstentryfmt</code>	50, 51, 73	<code>\Glsentryuseriv</code>	134
<code>\Glsentryfull</code>	136	<code>\glstentryuseriv</code>	134
<code>\glstentryfull</code>	136	<code>\Glsentryuserv</code>	135
<code>\Glsentryfullpl</code>	136	<code>\glstentryuserv</code>	134
<code>\glstentryfullpl</code>	136	<code>\Glsentryuservi</code>	135
<code>\glstentryitem</code>	162	<code>\glstentryuservi</code>	135
<code>\Glsentrylong</code>	135	<code>\glsexpandfields</code>	56
<code>\glstentrylong</code>	135	<code>\GLSfirst</code>	104
<code>\glstentrylongaccess</code>	278	<code>\Glsfirst</code>	104
<code>\Glsentrylongpl</code>	136	<code>\glsfirst</code>	103
<code>\glstentrylongpl</code>	135	<code>\glsfirstaccessdisplay</code>	279

<code>\GLSfirstplural</code>	107	<code>\Glsname</code>	108
<code>\Glsfirstplural</code>	106	<code>\glsname</code>	107, 108
<code>\glsfirstplural</code>	106, 107	<code>\glsnameaccessdisplay</code>	279
<code>\glsfirstpluralaccessdisplay</code>	279	<code>\glsnamefont</code>	169
<code>\glsgenentryfmt</code>	76	<code>\glsnavhyperlink</code>	210
<code>\glsgetgrouplabel</code>	167	<code>\glsnavhypertarget</code>	210
<code>\glsgetgrouptitle</code>	139, 166	<code>\glsnavigation</code>	211
<code>\glsglossarymark</code>	9, 32	<code>\glsnextpages</code>	160
<code>\glsgroupheading</code>	166, 168	<code>\glsnoexpandfields</code>	56
<code>\glsgroupskip</code>	166, 168, 214	<code>\glsnonextpages</code>	160
<code>\glshyperlink</code>	137	<code>\glsnoxindywarning</code>	35
<code>\glshypernavsep</code>	211	<code>\glsnumberformat</code>	30
<code>\glshyphnumber</code>	30, 169	<code>\glsnumbersgroupname</code>	26, 139, 166
<code>\glsIfListOfAcronyms</code>	14	<code>\glsnumlistlastsep</code>	137
<code>\glsinlinedescformat</code>	214	<code>\glsnumlistsep</code>	137
<code>\glsinlinedopostchild</code>	212, 213	<code>\glsopenbrace</code>	139
<code>\glsinlineemptydescformat</code>	214	<code>\glsorder</code>	21
<code>\glsinlinenameformat</code>	214	<code>\glsorg@endtheglossary</code>	5
<code>\glsinlineparentchildseparator</code>	213	<code>\glsorg@glossary</code>	4
<code>\glsinlinepostchild</code>	213	<code>\glsorg@theglossary</code>	5
<code>\glsinlineseparator</code>	213	<code>\glsorg@wrglossary</code>	4
<code>\glsinlinesubdescformat</code>	214	<code>\glspagelistwidth</code>	218, 224, 233, 240
<code>\glsinlinesubnameformat</code>	214	<code>\glspar</code>	28
<code>\glsinlinesubseparator</code>	213	<code>\GLSpl</code>	99
<code>\glskeylisttok</code>	176	<code>\Glspl</code>	98, 207
<code>\glslabeltok</code>	176	<code>\glspl</code>	72, 96, 98, 99
<code>\glslink</code>	72, 81, 92, 137, 167, 169	<code>\GLSplural</code>	105
<code>\glslink options</code>		<code>\Glsplural</code>	105
counter	80, 92, 200	<code>\glsplural</code>	104, 105
format	80, 92, 169	<code>\glspluralaccessdisplay</code>	279
hyper	80, 92	<code>\glspluralsuffix</code>	26, 51
local	81	<code>\glspostdescription</code>	8
<code>\glslistdottedwidth</code>	217	<code>\glspostinline</code>	214
<code>\glslocalreset</code>	70	<code>\glsprestandardsort</code>	10
<code>\glslocalresetall</code>	71	<code>\glsquote</code>	139
<code>\glslocalunset</code>	71	<code>\glsrefentry</code>	161
<code>\glslocalunsetall</code>	72	<code>\glsreset</code>	70
<code>\glslongaccessdisplay</code>	281	<code>\glsresetall</code>	71
<code>\glslongaccesskey</code>	293	<code>\glsresetentrylist</code>	160
<code>\glslongkey</code>	173	<code>\glsresetsubentrycounter</code>	161
<code>\glslongpluralaccessdisplay</code>	281	<code>\glssee</code>	154
<code>\glslongpluralaccesskey</code>	293	<code>\glsseeformat</code>	142, 154
<code>\glslongpluralkey</code>	173	<code>\glsseeitem</code>	155
<code>\glslongtok</code>	176	<code>\glsseeitemformat</code>	155
<code>\glsmakefirsttuc</code>	208	<code>\glsseelastsep</code>	154
<code>\glsmcols</code>	229	<code>\glsseelist</code>	154
<code>\glsmoveentry</code>	68	<code>\glsseesep</code>	154
<code>\GLSname</code>	108	<code>\glsSetAlphaCompositor</code>	30
		<code>\glsSetCompositor</code>	29

`\ifglstranslate` 19
`\ifglused` 45, 70
`\ifglxindy` 21
`index (style)` 245
`indexgroup (style)` 246
`indexhypergroup (style)` 247
`indexonlyfirst (option)` 20
`inline (style)` 212
`\inputencodingname` 21
`\istfile` 148
`\istfilename` 29
`\item` 169, 214, 245, 246

L

`link text` 72
`list (style)` 214
`listdotted (style)` 216
`listgroup (style)` 215
`listhypergroup (style)` 215
`\loadglsentries` 12, 72
`long (key)` 53
`long (style)` 218
`long3col (style)` 219
`long3colborder (style)` 220
`long3colheader (style)` 220
`long3colheaderborder (style)` .. 220
`long4col (style)` 220
`long4colborder (style)` 221
`long4colheader (style)` 221
`long4colheaderborder (style)` .. 222
`longaccess (key)` 275
`longborder (style)` 218
`longheader (style)` 219
`longheaderborder (style)` 219
`\longnewglossaryentry` 50, 62
`longplural (key)` 54
`longpluralaccess (key)` 275
`\longprovideglossaryentry` ... 62
`longragged (style)` 224
`longragged3col (style)` 225
`longragged3colborder (style)` .. 226
`longragged3colheader (style)` .. 226
`longragged3colheaderborder (style)` 226
`longraggedborder (style)` 224
`longraggedheader (style)` 225
`longraggedheaderborder (style)` 225
`longtable (environment)`
..... 8, 195, 217–228

`longtable package` 217, 223

M

`\makefirstuc` 207
`makeglossaries` .. 21, 29, 42, 48, 157
`\makeglossaries`
..... 25, 29, 30, 47, 49, 147, 148
`\makeglossary` 148
`makeindex` 307
`makeindex` 10, 21, 26, 29–
31, 48–50, 69, 84, 87, 139, 142,
144, 146, 152, 165, 166, 254, 255
`delim_n` 30
`delim_r` 31
`page_compositor` 29
`special characters` 85, 86, 139
`makeindex (option)` 21
`mcolalttree (style)` 231
`mcolalttreegroup (style)` 232
`mcolalttreehypergroup (style)` . 232
`mcolindex (style)` 229
`mcolindexgroup (style)` 229
`mcolindexhypergroup (style)` ... 229
`mcoltree (style)` 230
`mcoltreegroup (style)` 230
`mcoltreehypergroup (style)` 230
`mcoltreenoname (style)` 231
`mcoltreenonamegroup (style)` ... 231
`mcoltreenonamehypergroup (style)` 231
`memoir class` 149
`mfistuc package` 1
`\mfistucMakeUppercase` 208
`multicol package` 229
`multicols (environment)` 229

N

`name (key)` 50
`\new@glossaryentry` 57
`\newacronym` 20, 53, 54, 72, 172
`\newacronymhook` 176
`\newglossary` 15, 48, 49, 146, 147, 158
`\newglossaryentry` .. 50, 56, 72, 172
`\newglossaryentry options`
`access` 276, 277
`counter` 52
`description` 20, 50,
54, 56, 63, 109, 132, 173, 186, 274
`descriptionaccess` 278, 280
`descriptionplural` 110, 275

- descriptionpluralaccess .. 278, 280
- first 51,
65, 92, 103, 133, 183, 188, 189, 274
- firstaccess 277, 279
- firstplural 51, 106, 133, 274
- firstpluralaccess 277, 279
- format 141
- long 135, 275
- longaccess 278, 281
- longplural 135, 275
- longpluralaccess 278, 281
- name 50,
51, 54, 56, 63, 107, 131, 155, 274
- nonumberlist 52
- parent 52, 56
- plural 51, 65, 104, 274
- pluralaccess 277, 279
- prefix 201
- prefixfirst 201
- prefixfirstplural 201
- prefixplural 201
- see 7, 52, 147
- short 135, 275
- shortaccess 278, 281
- shortplural 135, 275
- shortpluralaccess 278, 281
- sort 50, 134, 165, 166
- symbol ... 50, 51, 112, 133, 179–
181, 183, 188, 220, 236, 274, 276
- symbolaccess 277, 280
- symbolplural 113, 274
- symbolpluralaccess 278, 280
- text . 51, 92, 102, 132, 179, 183, 274
- textaccess 277, 279
- type 12, 52, 72, 133
- user1 115, 134, 275
- user2 116, 134
- user3 117, 134
- user4 119, 134
- user5 120, 134
- user6 122, 135, 275
- \newglossarystyle 168
- nogroupskip (option) 9
- nohypertypes (option) 15
- \noist 145, 200, 259
- nolist (option) 8
- nolong (option) 8
- nomain (option) 12
- nonumberlist (key) 52

- nonumberlist (option) 7
- \nopostdesc 28
- 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 172
- order (option) 21

P

- package options:
 - acronym 12, 13, 25, 158, 172
 - true 13
 - acronym 13
 - acronymlists 15
 - acronyms 13
 - compatible-2.07 22
 - compatible-3.07 22
 - counter 15
 - counter 15
 - description 183, 184
 - description 20
 - dua 182–184
 - dua 20
 - entrycounter 160
 - true 9
 - entrycounter 9
 - entrycounterwithin 9
 - footnote 93,
95–97, 99–101, 180, 182, 183, 186
 - footnote 20
 - hyperfirst
 - false 93, 95–97, 99–101
 - hyperfirst 20
 - indexonlyfirst 314
 - indexonlyfirst 20
 - makeindex 142, 200
 - makeindex 21
 - nogroupskip 9
 - nohypertypes 15
 - nolist 195
 - nolist 8
 - nolong 195, 217

\SetDefaultAcronymDisplayStyle	\showgloiname	197
..... 176	\showgloinameaccess	293
\SetDefaultAcronymStyle 177	\showgloparent	195
\SetDescriptionAcronymDisplayStyle	\showgloplural	196
..... 182	\showglopluralaccess	293
\SetDescriptionAcronymStyle 183	\showgloshort	198
\SetDescriptionDUAAcronymDisplayStyle	\showgloshortaccess	294
..... 180	\showgloshortpluralaccess ..	294
\SetDescriptionDUAAcronymStyle	\showglosort	198
..... 181	\showglossaries	199
\SetDescriptionFootnoteAcronymDisplayStyle	\showglossarycounter	200
..... 178	\showglossaryentries	200
\SetDescriptionFootnoteAcronymStyle	\showglossaryin	199
..... 179	\showglossaryout	199
\SetDUADisplayStyle 189	\showglossarytitle	200
\SetDUASStyle 190	\showglosymbol	198
\setentrycounter 167	\showglosymbolaccess	293
\SetFootnoteAcronymDisplayStyle	\showglosymbolplural	198
..... 184	\showglosymbolpluralaccess .	293
\SetFootnoteAcronymStyle ... 186	\showglotext	196
\setglossarypreamble 31	\showglotextaccess	293
\setglossarysection 33	\showglotype	196
\setglossarystyle 167	\showglouserii	197
\setglossentrycompatibility 165	\showglouseriii	197
\SetSmallAcronymDisplayStyle 187	\showglouseriv	197
\SetSmallAcronymStyle 188	\showglouserv	197
\setStyleFile 28, 29	\showglouservi	197
\setupglossaries 23	smallcaps (option)	20
short (key) 53	smaller (option)	20
shortaccess (key) 275	\SmallNewAcronymDef	187, 292
shortplural (key) 53	sort (key)	50
shortpluralaccess (key) 275	sort (option)	9
shotcuts (option) 21	style (option)	6
\showacronymlists 199	subentrycounter (option)	9
\showglocounter 196	\subglossentry	163
\showglodesc 198	\subitem	246
\showglodescaccess 294	sublistdotted (style)	217
\showglodescplural 198	\subsubitem	246
\showglodescpluralaccess ... 294	super (style)	233
\showglofirst 196	super3col (style)	235
\showglofirstaccess 293	super3colborder (style)	235
\showglofirsttpl 196	super3colheader (style)	236
\showglofirstpluralaccess .. 293	super3colheaderborder (style)	236
\showgloflag 199	super4col (style)	236
\showgloindex 199	super4colborder (style)	237
\showglolevel 195	super4colheader (style)	237
\showglolevel 198	super4colheaderborder (style)	238
\showglolevelaccess 294	superborder (style)	234
\showglolevelpluralaccess ... 294		

superheader (style)	234
superheaderborder (style)	234
superragged (style)	240
superragged3col (style)	241
superragged3colborder (style) .	242
superragged3colheader (style) .	242
superraggedborder (style)	240
superraggedheader (style)	241
superraggedheaderborder (style)	241
superraggedright3colheaderborder (style)	243
supertabular (environment) ...	
.....	8, 195, 232–245
supertabular package ..	8, 195, 233, 239
symbol (key)	51
symbolaccess (key)	274
\symbolname	26
symbolplural (key)	51
symbolpluralaccess (key)	274
symbols (option)	22
T	
text (key)	51
textaccess (key)	274
textcase package	3
\theequation	152
theglossary (environment)	
.....	5, 31, 162,
.....	163, 168, 230, 231, 247, 248, 250
\theHequation	153
theindex (environment)	245
toc (option)	5
\translate	26
translate (option)	19

translator package	
... ..	24, 26, 27, 155, 294, 303–307
tree (style)	247
treegroup (style)	248
treehypergroup (style)	248
treenoname (style)	248
treenonamegroup (style)	249
treenonamehypergroup (style) ..	249
type (key)	52

U

ucmark (option)	9
user1 (key)	53
user2 (key)	53
user3 (key)	53
user4 (key)	53
user5 (key)	53
user6 (key)	53

W

\warn@nomakeglossaries	146
\warn@noprintglossary	155
\writeist	29, 36, 37, 40, 140, 253, 255

X

\xcapitalisewords	209
\xglsaccsupp	279
xindy	307
xindy	10, 21, 22, 29,
.....	35, 38, 40–43, 69, 90, 139, 140,
.....	142, 152, 157, 165, 200, 254, 255
xindy (option)	22
xindygloss (option)	22
\xmakefirstuc	209
xspace package	4, 172