glossaries-extra.sty v1.49: an extension to the glossaries package

Nicola L.C. Talbot
Dickimaw Books
dickimaw-books.com
2022-10-14

This document is also available as HTML (glossaries-extra-manual.html).

Abstract

The glossaries-extra package is an extension to the glossaries package, providing additional features. In particular, it provides a completely different abbreviation mechanism. You will need at least glossaries version 4.19, but it is best to update both packages at the same time, if new releases are available for both of them.

The glossaries-extra package uses a different set of defaults to the base glossaries package. This means that if you simply replace glossaries with glossaries-extra in an existing document, there may be some differences in the PDF, and you may encounter errors. See §1.1 for more details.

This document assumes some familiarity with the glossaries package. If you are new to glossaries, you may prefer to start with the following:

- The glossaries package: a guide for beginners
  
  texdoc glossariesbegin

- glossaries-extra and bib2gls: an introductory guide
texdoc bib2gls-begin
Contents

List of Examples vii

I. User Guide 1

1. Introduction 2
   1.1. Package Defaults 2
   1.2. Example Differences Between glossaries and glossaries-extra 3
      1.2.1. Basic defaults 3
      1.2.2. Language defaults 4
      1.2.3. Combined with memoir 4
      1.2.4. Abbreviations 5
      1.2.5. Glossary Mid-Build Placeholder (\printglossary) 5
   1.3. Further Reading 7

2. Package Options 8
   2.1. Glossary Lists 9
   2.2. Glossary Style Options 12
   2.3. Loading Other Packages 14
   2.4. Entry Definitions, References and Indexing 15
   2.5. Debugging 28

3. Defining Entries 32
   3.1. Command Definitions 32
   3.2. Glossary Entry Keys 33
   3.3. Plurals 35
   3.4. Entry Aliases 36
   3.5. Setting or Updating Fields 37

4. Abbreviations 41
   4.1. Defining Abbreviations 41
      4.1.1. Abbreviation Fields: long and short 43
      4.1.2. Abbreviation Fields: longplural and shortplural 43
      4.1.3. Abbreviation Fields: name and description 44
      4.1.4. Abbreviation Fields: type 44
      4.1.5. General Hooks 45
   4.2. Examples: makeindex vs bib2gls 45
## 4. Referencing (Using) Abbreviations
- 4.3. Referencing (Using) Abbreviations ........................................ 47
  - 4.3.1. Prefixes ................................................................. 54
  - 4.3.2. Abbreviation Shortcut Commands .................................. 56
- 4.4. Tagging Initials ............................................................. 56
- 4.5. Abbreviation Styles ....................................................... 59
  - 4.5.1. Predefined Abbreviation Styles .................................. 60
  - 4.5.2. Advanced Style Commands ...................................... 159
  - 4.5.3. Defining New Abbreviation Styles ............................... 161
- 4.6. Restoring Base Acronym Mechanism ................................. 181

## 5. Referencing (Using) Entries ........................................... 183
- 5.1. Options ........................................................................... 186
  - 5.1.1. Setting Up Defaults .................................................. 187
  - 5.1.2. Additional Options .................................................. 190
- 5.2. Case Changing .............................................................. 196
  - 5.2.1. Sentence Case Commands ....................................... 197
  - 5.2.2. Lower Case ............................................................ 200
  - 5.2.3. Upper Case ............................................................ 200
  - 5.2.4. Title Case .............................................................. 200
- 5.3. Entries in Sectioning Titles, Headers, Captions and Contents .... 200
  - 5.3.1. Simplistic Approach ................................................ 201
  - 5.3.2. New Commands Designed for Chapter/Section Headings or Captions 204
  - 5.3.3. Advanced Commands ............................................. 214
- 5.4. Nested Links ................................................................. 228
- 5.5. Adjusting the Text Style ................................................ 234
  - 5.5.1. Outer Formatting .................................................... 238
  - 5.5.2. Middle Formatting .................................................. 239
  - 5.5.3. Inner Formatting .................................................... 241
  - 5.5.4. Post Link Hook ....................................................... 245
  - 5.5.5. Entry Format Style ................................................ 254
- 5.6. Hyperlinks ...................................................................... 256
- 5.7. Label Prefixes ............................................................... 256
- 5.8. Indexing ........................................................................... 259
- 5.9. Cross-Referencing ........................................................... 265
  - 5.9.1. Entries that may not be required ................................ 267
  - 5.9.2. Accessing the Cross-Referencing Fields ....................... 276
  - 5.9.3. Cross-Reference Indexing ...................................... 278
- 5.10. First Use Flag .................................................................. 279
  - 5.10.1. Buffering Unsets .................................................... 284
- 5.11. Accessing Fields .......................................................... 291
- 5.12. Encapsulation (Formatting) Based on Field Values .............. 293
  - 5.12.1. Foreign Language Field ........................................... 293
  - 5.12.2. Associated Entry Format ........................................ 295
- 5.13. Comma-Separated Lists ................................................. 299
5.14. List Fields .................................................. 304
5.15. Field Conditionals ........................................... 305

6. Counting References ............................................. 307
  6.1. Entry Counting (First Use Flag) ........................... 307
  6.2. Link Counting ................................................. 317

7. Multi (or Compound) Entries .................................... 322
  7.1. Examples ....................................................... 326
      7.1.1. Example: Hierarchical ................................. 326
      7.1.2. Example: Suffix ......................................... 327
      7.1.3. Example: Category Suffix .............................. 328
      7.1.4. Example: Separators .................................... 329
      7.1.5. Example: Skipping Elements (Fragment Element) ... 330
      7.1.6. Example: Skipping Elements (Prefix and Post-Link Hooks) 333
  7.2. Main and Other Elements .................................... 339
  7.3. Prefixes and Suffixes ....................................... 340
  7.4. Separators .................................................... 343
  7.5. \gls Element Hooks ......................................... 347
  7.6. Post-Link Hook ................................................ 349
      7.6.1. Last Element ............................................. 350
      7.6.2. Main Element ............................................ 351
  7.7. Multi-Entry First Use ........................................ 352
  7.8. Multi-Entry Category ......................................... 353
  7.9. Multi-Entry Settings ......................................... 354
      7.9.1. Indexing .................................................. 355
      7.9.2. Location Formats (Encaps) ............................. 355
      7.9.3. Post-Link Hooks .......................................... 356
      7.9.4. Prefixes and Suffixes ................................. 357
      7.9.5. Skipping Elements ..................................... 357
      7.9.6. General ................................................... 358
  7.10. \gls Options ................................................ 359
  7.11. Variants of \gls ............................................. 363
      7.11.1. \glb-like ................................................. 363
      7.11.2. Abbreviations .......................................... 364
      7.11.3. Other Fields ............................................ 365
      7.11.4. Support for glossaries-prefix (\pgls) .............. 367
  7.12. Cross-References ............................................ 369
  7.13. Additional Commands ........................................ 370
  7.14. bib2glb ....................................................... 372

8. Defining and Displaying Glossaries ................................. 375
  8.1. Entry Page Reference ....................................... 377
  8.2. Glossary Preamble ............................................ 377
### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.3</td>
<td>Options</td>
<td>377</td>
</tr>
<tr>
<td>8.4</td>
<td>Displaying a Glossary Without Sorting or Indexing</td>
<td>382</td>
</tr>
<tr>
<td>8.4.1</td>
<td>Groups and Hierarchy</td>
<td>387</td>
</tr>
<tr>
<td>8.4.2</td>
<td>Location Lists</td>
<td>397</td>
</tr>
<tr>
<td>8.4.3</td>
<td>Advanced Commands</td>
<td>398</td>
</tr>
<tr>
<td>8.5</td>
<td>Standalone Entry Items</td>
<td>412</td>
</tr>
<tr>
<td>8.6</td>
<td>Glossary Style Modifications</td>
<td>418</td>
</tr>
<tr>
<td>8.6.1</td>
<td>Post-Name Hooks</td>
<td>420</td>
</tr>
<tr>
<td>8.6.2</td>
<td>Post-Description Hooks</td>
<td>421</td>
</tr>
<tr>
<td>8.6.3</td>
<td>Number (Location) List</td>
<td>423</td>
</tr>
<tr>
<td>8.6.4</td>
<td>Indexing Groups</td>
<td>425</td>
</tr>
<tr>
<td>8.6.5</td>
<td>glossaries-extra-stylemods</td>
<td>427</td>
</tr>
<tr>
<td>8.7</td>
<td>New Glossary Styles</td>
<td>437</td>
</tr>
<tr>
<td>8.7.1</td>
<td>glossary-bookindex package</td>
<td>437</td>
</tr>
<tr>
<td>8.7.2</td>
<td>glossary-longextra package</td>
<td>444</td>
</tr>
<tr>
<td>8.7.3</td>
<td>glossary-topic package</td>
<td>461</td>
</tr>
<tr>
<td>8.7.4</td>
<td>glossary-table package</td>
<td>468</td>
</tr>
<tr>
<td>9.</td>
<td>Accessibility Support</td>
<td>475</td>
</tr>
<tr>
<td>9.1</td>
<td>Abbreviations</td>
<td>475</td>
</tr>
<tr>
<td>9.2</td>
<td>Accessibility Wrappers</td>
<td>477</td>
</tr>
<tr>
<td>9.3</td>
<td>Inner Formatting Wrappers</td>
<td>484</td>
</tr>
<tr>
<td>10.</td>
<td>Categories</td>
<td>492</td>
</tr>
<tr>
<td>10.1</td>
<td>Known Categories</td>
<td>493</td>
</tr>
<tr>
<td>10.2</td>
<td>Attributes</td>
<td>494</td>
</tr>
<tr>
<td>10.2.1</td>
<td>Known Attributes</td>
<td>494</td>
</tr>
<tr>
<td>10.2.2</td>
<td>Accessing and Setting Attributes</td>
<td>504</td>
</tr>
<tr>
<td>11.</td>
<td>bib2gls: Managing Reference Databases</td>
<td>508</td>
</tr>
<tr>
<td>11.1</td>
<td>Indexing (Recording)</td>
<td>513</td>
</tr>
<tr>
<td>11.2</td>
<td>Selection</td>
<td>513</td>
</tr>
<tr>
<td>11.3</td>
<td>Sorting and Displaying the Glossary</td>
<td>514</td>
</tr>
<tr>
<td>11.4</td>
<td>Record Counting</td>
<td>522</td>
</tr>
<tr>
<td>11.4.1</td>
<td>Unit Record Counting</td>
<td>528</td>
</tr>
<tr>
<td>11.4.2</td>
<td>Mini-Glossaries</td>
<td>531</td>
</tr>
<tr>
<td>11.5</td>
<td>The glossaries-extra-bib2gls package</td>
<td>533</td>
</tr>
<tr>
<td>11.5.1</td>
<td>Displaying Glossaries</td>
<td>534</td>
</tr>
<tr>
<td>11.5.2</td>
<td>Helper Commands for Resource Options</td>
<td>535</td>
</tr>
<tr>
<td>11.5.3</td>
<td>Commands Used Within Resource Files</td>
<td>553</td>
</tr>
<tr>
<td>11.5.4</td>
<td>Hierarchy</td>
<td>554</td>
</tr>
<tr>
<td>11.5.5</td>
<td>Supplemental Locations</td>
<td>554</td>
</tr>
<tr>
<td>11.5.6</td>
<td>Nameref Records</td>
<td>555</td>
</tr>
<tr>
<td>11.5.7</td>
<td>Dual Entry Commands</td>
<td>560</td>
</tr>
</tbody>
</table>
11.5.8. Supplementary Commands .............................................. 571

12. Auto-Indexing ................................................................. 574

13. On-the-Fly Document Definitions ........................................ 578

14. Supplementary Files ...................................................... 581
   14.1. Dummy Files for Testing ........................................... 581
   14.2. Sample Files ......................................................... 582

15. Multi-Lingual Support ...................................................... 592

II. Summaries and Index .......................................................... 597

Symbols ................................................................................. 598

Terms ..................................................................................... 599

Glossary Entry Keys Summary ................................................. 605

Glossary Entry Fields Summary .............................................. 614

\gls-Like and \glstext-Like Options Summary .............................. 616

Multi-Entry Set Options Summary .......................................... 619

Print [Unsrt|noidx] Glossary Options Summary ......................... 624

Abbreviation Styles Summary .................................................. 628

Glossary Styles Summary ....................................................... 645

Command Summary ............................................................... 652
   Command Summary: @ ........................................................ 652
   Command Summary: A ....................................................... 653
   Command Summary: B ....................................................... 663
   Command Summary: C ....................................................... 664
   Command Summary: D ....................................................... 667
   Command Summary: E ....................................................... 671
   Command Summary: F ....................................................... 672
   Command Summary: G ....................................................... 673
   Command Summary: Glo .................................................... 674
   Command Summary: Gls .................................................... 677
   Command Summary: Glsxtr ............................................... 779
   Command Summary: H ....................................................... 894
List of Examples

1. Multiple abbreviation styles ........................................ 42
2. \newabbreviation vs \newacronym vs \newglossaryentry .............. 46
3. @abbreviation vs @acronym vs @entry .......................... 47
4. Referencing an abbreviation (with hyperref) .................... 49
5. First use of \gls vs \glsxtrfull vs \glsfirst .......................... 51
6. Tagging abbreviation initials ....................................... 59
7. Category without an associated abbreviation style .............. 60
8. The short-nolong abbreviation style .............................. 65
9. The short-nolong-desc abbreviation style ......................... 66
10. The nolong-short abbreviation style .............................. 67
11. The short-sc-nolong abbreviation style ........................... 68
12. The short-sc-nolong-desc abbreviation style ..................... 68
13. The nolong-short-sc abbreviation style .......................... 69
14. The short-sm-nolong abbreviation style ........................... 70
15. The short-sm-nolong-desc abbreviation style ..................... 70
16. The nolong-short-sm abbreviation style ........................... 71
17. The short-em-nolong abbreviation style ........................... 72
18. The short-em-nolong-desc abbreviation style ..................... 72
19. The nolong-short-em abbreviation style ........................... 73
20. The long-noshort-desc abbreviation style ......................... 74
21. The long-noshort abbreviation style ............................. 75
22. The long-noshort-sc abbreviation style ........................... 76
23. The long-noshort-sc-desc abbreviation style ..................... 76
24. The long-noshort-sm abbreviation style ........................... 77
25. The long-noshort-sm-desc abbreviation style ..................... 78
26. The long-noshort-em abbreviation style ........................... 78
27. The long-noshort-em-desc abbreviation style ..................... 79
28. The long-em-noshort-em abbreviation style ....................... 80
29. The long-em-noshort-em-desc abbreviation style ................. 80
30. The long-short abbreviation style .................................. 81
31. The long-short-desc abbreviation style ........................... 82
32. The long-short-sc abbreviation style .............................. 83
33. The long-short-sc-desc abbreviation style ......................... 84
34. The long-short-sm abbreviation style .............................. 84
35. The long-short-sm-desc abbreviation style ......................... 85
36. The long-short-em abbreviation style .............................. 86
List of Examples

37. The long-short-em-desc abbreviation style .......... 86
38. The long-em-short-em abbreviation style ............ 87
39. The long-em-short-em-desc abbreviation style ....... 88
40. The long-short-user abbreviation style .............. 89
41. The long-short-user-desc abbreviation style ......... 89
42. The long-postshort-user abbreviation style ......... 90
43. The long-postshort-user-desc abbreviation style ... 91
44. The long-postshort-sc-user abbreviation style ... 92
45. The long-postshort-sc-user-desc abbreviation style ... 92
46. The short-long abbreviation style ................. 93
47. The short-long-desc abbreviation style .......... 94
48. The short-sc-long abbreviation style .............. 95
49. The short-sc-long-desc abbreviation style ....... 95
50. The short-sm-long abbreviation style .......... 96
51. The short-sm-long-desc abbreviation style ....... 97
52. The short-em-long abbreviation style .............. 97
53. The short-em-long-desc abbreviation style ....... 98
54. The short-em-long-em abbreviation style ....... 99
55. The short-em-long-em-desc abbreviation style .... 99
56. The short-long-user abbreviation style .......... 100
57. The short-long-user-desc abbreviation style ...... 101
58. The short-postlong-user abbreviation style ....... 102
59. The short-postlong-user-desc abbreviation style ... 103
60. The long-hyphen-short-hyphen abbreviation style ... 104
61. The long-hyphen-postshort-hyphen abbreviation style ... 105
62. The long-hyphen-short-hyphen-desc abbreviation style ... 106
63. The long-hyphen-postshort-hyphen-desc abbreviation style ... 106
64. The long-hyphen-noshort-desc-noreg abbreviation style ... 107
65. The long-hyphen-noshort-noreg abbreviation style ... 108
66. The short-hyphen-long-hyphen abbreviation style ... 109
67. The short-hyphen-postlong-hyphen abbreviation style ... 109
68. The short-hyphen-long-hyphen-desc abbreviation style ... 110
69. The short-hyphen-postlong-hyphen-desc abbreviation style ... 111
70. The long-only-short-only abbreviation style ....... 112
71. The long-only-short-only-desc abbreviation style ... 112
72. The long-only-short-sc-only abbreviation style ... 113
73. The long-only-short-sc-only-desc abbreviation style ... 114
74. The short-footnote abbreviation style ............ 115
75. The short-footnote-desc abbreviation style ....... 116
76. The short-postfootnote abbreviation style ....... 117
77. The short-postfootnote-desc abbreviation style ... 118
78. The short-sc-footnote abbreviation style .......... 119
79. The short-sc-footnote-desc abbreviation style .... 120
80. The short-sc-postfootnote abbreviation style .... 121
List of Examples

81. The short-sc-postfootnote-desc abbreviation style .......................................... 122
82. The short-sm-footnote abbreviation style ............................................................ 123
83. The short-sm-footnote-desc abbreviation style ..................................................... 124
84. The short-sm-postfootnote abbreviation style ....................................................... 125
85. The short-sm-postfootnote-desc abbreviation style .............................................. 126
86. The short-em-footnote abbreviation style ............................................................ 127
87. The short-em-footnote-desc abbreviation style ..................................................... 128
88. The short-em-postfootnote abbreviation style ....................................................... 129
89. The short-em-postfootnote-desc abbreviation style .............................................. 130
90. Illustrating the prereset option .......................................................................... 192
91. Combining prereset and preunset ........................................................................ 193
92. References in section headings (simplistic approach) .......................................... 203
93. References in section headings using \glsfmttext .................................................... 205
94. Reference with hyperlink in section headings ....................................................... 206
95. Nested link text with \glsp1 ................................................................................ 233
96. Link text styles: outer, middle, inner, hyperlinks and post-link hook ..................... 236
97. Link text styles: outer, middle, inner, hyperlinks and post-link hooks (custom and abbreviation style) ................................................................. 238
98. Changing the outer text format .......................................................................... 239
99. Middle formatting ............................................................................................... 240
100. Inner formatting ................................................................................................ 242
101. Category post-link hook .................................................................................. 246
102. Category post-link hook with punctuation lookahead ......................................... 249
103. Category post-link hooks .................................................................................. 252
104. Location list ordering (makeindex) .................................................................... 262
105. Location list ordering (bib2gls) ........................................................................... 263
106. Cross-references (autoseeindex=true) ............................................................... 268
107. Cross-references (autoseeindex=false) ............................................................... 269
108. Cross-references (autoseeindex=false and post-name hook) .............................. 270
109. Cross-references (no see, seealso or alias) ......................................................... 272
110. Cross-references (bib2gls) .................................................................................. 274
111. Cross-references (bib2gls and selection=recorded and deps and see) ............... 275
112. Cross-references (bib2gls and selection=recorded and deps and see, prune-xr) ................................................................. 276
113. Resetting the first use flag (\glsreset) ................................................................. 282
114. Local unset ....................................................................................................... 283
115. Abbreviations with beamer (unset buffering) ..................................................... 288
116. Buffering first use unsets with \mbox ................................................................. 289
117. Alternatives to buffering .................................................................................... 290
118. Foreign language field encapsulation ................................................................ 295
119. Storing a formatting command in a field ............................................................ 298
120. Formatting lists contained in field values ........................................................... 303
121. Entry counting according to category ................................................................ 312
122. Entry unit counting (per section) according to category .................................... 315
List of Examples

123. Enabling unit counting to hook into hyperlink setting .................. 317
124. Link counting used to selectively suppress hyperlinks .................. 320
125. Multi-entries: hierarchical ........................................... 327
126. Multi-entries: hierarchical with first-use suffix ......................... 328
127. Multi-entries: hierarchical with category suffix ........................ 329
128. Multi-entries: separators ................................................. 330
129. Multi-entries: skipping elements ....................................... 332
130. Multi-entries: skipping elements (unsetting others) ...................... 333
131. Multi-entries: skipping elements (prefix and post-link hooks) ......... 337
132. Changing the target prefix .............................................. 380
133. Prepending to the target prefix for just the entry item .................. 381
134. Displaying unsorted glossaries ........................................ 384
135. Displaying unsorted glossaries with stylemods .......................... 385
136. Displaying unsorted glossaries with different group settings .......... 386
137. Displaying unsorted glossaries with a copied list ....................... 388
138. Displaying unsorted glossaries with custom groups ..................... 392
139. Displaying unsorted glossaries with custom groups and sub-group headings 393
140. Displaying sorted glossaries with groups using bib2gls .................. 397
141. Filtering by category .................................................... 403
142. Inner glossaries using printunsrtglossarywrap .......................... 405
143. Nested glossaries ....................................................... 406
144. Sub-glossary for a given counter value ................................ 411
145. Sub-glossary for a given counter value ordered by use in the section ... 413
146. The bookindex style ..................................................... 438
147. The topic style ........................................................... 462
148. The topicmcols style ...................................................... 463
149. The topicmcols style with the widest name set .......................... 464
150. Two entries per row with \printunsrttable ................................. 474
151. Using bib2gls: one resource set ....................................... 516
152. Using bib2gls: multiple resource sets ................................... 518
153. Using bib2gls: sub-blocks ............................................... 520
154. Using bib2gls: record counting ........................................ 528
155. Using bib2gls: unit record counting ..................................... 531
156. Using bib2gls: unit record counting mini-glossary ....................... 533
157. Using bib2gls: simple custom sort rule .................................. 538
158. Using bib2gls: dual backlinks .......................................... 562
159. Using bib2gls: dual entry label prefixes ................................ 570
Part I.

User Guide
1. Introduction

The glossaries package is a flexible package, but it’s also a heavy-weight package that uses a lot of resources. As package developer, I’m caught between those users who complain about the drawbacks of a heavy-weight package with a large user manual and those users who want more features (which necessarily adds to the package weight and manual size).

The glossaries-extra package is an attempt to provide a compromise for this conflict. Version 4.22 of the glossaries package is the last version to incorporate any major new features. Future versions of glossaries will mostly just be bug fixes. New features will instead be added to glossaries-extra. This means that the base glossaries package won’t increase in terms of package loading time and allocation of resources, but those users who do want extra features available will have more of a chance of getting their feature requests accepted.

Some of the commands provided by the base glossaries package are incompatible with glossaries-extra. These are marked with ☞ in this document.

The glossaries-extra package internally loads the glossaries package. As a general rule, it’s better to defer loading the base glossaries package to glossaries-extra rather than loading the two packages separately.

1.1. Package Defaults

I’m not happy with some of the default settings assumed by the glossaries package, and, judging from code I’ve seen, other users also seem unhappy with them, as certain package options are often used in questions posted on various sites. I can’t change the default behaviour of glossaries as it would break backward compatibility, but since glossaries-extra is a separate package, I have decided to implement some of these commonly-used options by default. You can switch them back if they’re not appropriate.

The new defaults are:

- **toc=true** (add the glossaries to the table of contents). Use **toc=false** to switch this back off.

- **nopostdot=true** (suppress the terminating full stop after the description in the glossary). Use **nopostdot=false** or just **postdot** to restore the terminating full stop. Alternatively, if you are interested in switching to bib2gls, you can instruct bib2gls to insert it with the **post-description-dot** option.
1. Introduction

- `noredefwarn` (suppress the warnings that occur when the `theglossary` environment and `printglossary` are redefined while `glossaries` is loading). Note that this won’t have any effect if the `glossaries` package has already been loaded before you load the `glossaries-extra` package.

- If babel has been loaded, the `translate=babel` option is switched on. To revert to using the translator interface, use `translate=true`. There is no change to the default if babel hasn’t been loaded.

- The default style used by `\newacronym` is `short-nolong`. (That is, the long form is not shown on first use.) To revert back to “⟨long⟩ ⟨short⟩” on first use do:

\begin{verbatim}
\setabbreviationstyle[acronym]{long-short}
\end{verbatim}

In the above example, `long-short` refers to the `glossaries-extra` abbreviation style not the `glossaries` acronym style of the same name. See §4 for further details.

1.2. Example Differences Between glossaries and glossaries-extra

The examples below illustrate the difference in explicit package options between `glossaries` and `glossaries-extra`. There may be other differences resulting from modifications to commands provided by `glossaries`.

1.2.1. Basic defaults

\begin{verbatim}
\documentclass{article}
\usepackage[glossaries-extra]{glossaries-extra}
\end{verbatim}

This is like:

\begin{verbatim}
\documentclass{article}
\usepackage[toc,nopostdot]{glossaries}
\usepackage{glossaries-extra}
\end{verbatim}
1.2.2. Language defaults

\documentclass[british]{article}  \
\usepackage{babel}  \
\usepackage{glossaries-extra}

This is like:

\documentclass[british]{article}  \
\usepackage{babel}  \
\usepackage[toc,nopostdot,translate=babel]{glossaries}  \
\usepackage{glossaries-extra}

1.2.3. Combined with memoir

\documentclass{memoir}  \
\usepackage{glossaries-extra}

This is like:

\documentclass{memoir}  \
\usepackage[toc,nopostdot,noredefwarn]{glossaries}  \
\usepackage{glossaries-extra}

However

\documentclass{memoir}  \
\usepackage{glossaries}  \
\usepackage{glossaries-extra}

This is like:

\documentclass{memoir}  \
\usepackage[toc=false,nopostdot=false]{glossaries}  \
\usepackage{glossaries-extra}
1. Introduction

Since by the time glossaries-extra has been loaded, the base glossaries package has already redefined memoir’s glossary-related commands.

1.2.4. Abbreviations

Abbreviations are defined with \newabbreviation:

\usepackage{glossaries-extra}
\newabbreviation{svm}{SVM}{support vector machine}
\begin{document}
First use: \gls{svm}. Explicit full form: \glsxtrfull{svm}.
\end{document}

This is the closest match to:

\usepackage{glossaries}
\newacronym{svm}{SVM}{support vector machine}
\begin{document}
First use: \gls{svm}. Explicit full form: \acrfull{svm}.
\end{document}

If you want to continue using \newacronym then you will need to change the style for the acronym category:

\usepackage{glossaries-extra}
\setabbreviationstyle[acronym]{long-short}
\newacronym{svm}{SVM}{support vector machine}
\begin{document}
First use: \gls{svm}. Explicit full form: \glsxtrfull{svm}.
\end{document}

Don’t use commands like \glsfirst or \glstext with abbreviations. See §4 for further details.

1.2.5. Glossary Mid-Build Placeholder (\printglossary)

Another noticeable change with glossaries-extra is that by default \printglossary will now display information text in the document if the external glossary file doesn’t exist. This is explanatory text to help new users who can’t work out what to do next to complete the
document build. Once the document is set up correctly and the external files have been generated, this text will disappear.

This change is mostly likely to be noticed by users with one or more redundant empty glossaries who ignore transcript messages, explicitly use makeindex/xindy on just the non-empty glossary (or glossaries) and use the iterative \printglossaries command instead of \printglossary. For example, consider the following:

```
\documentclass{article}
\usepackage[acronym]{glossaries}
\makeglossaries
\newacronym{laser}{laser}{light amplification by stimulated emission of radiation}
\begin{document}
\gls{laser}
\printglossaries
\end{document}
```

The above document will only display the list of acronyms at the place where \printglossaries occurs. However it will also attempt to input the gls file associated with the main glossary.

If you use makeglossaries, you’ll get the warning message:

```
Warning: File 'test.glo' is empty.
Have you used any entries defined in glossary 'main'?
Remember to use package option 'nomain' if you
don't want to use the main glossary.
```

(where the original file is called test.tex) but if you simply call makeindex directly to generate the acr file (without attempting to create the gls file) then the transcript file will always contain the message:

```
No file test.gls.
```

This doesn’t occur with makeglossaries as it will create the gls file containing the single command \null.

If you simply change from glossaries to glossaries-extra in this document, you’ll find a change in the resulting PDF if you don’t use makeglossaries and you only generate the acr file with makeindex.

The transcript file will still contain the message about the missing gls, but now you’ll also see information in the actual PDF document. The simplest remedy is to follow the advice inserted into the document at that point, which is to add the nomain package option:
1. Introduction

\documentclass{article}
\usepackage[nomain,acronym,postdot]{glossaries-extra}
\makeglossaries
\setabbreviationstyle[acronym]{long-short}
\newacronym{laser}{laser}{light amplification by stimulated emission of radiation}
\begin{document}
\gls{laser}
\printglossaries
\end{document}

Note the need to set the acronym style using \setabbreviationstyle before \newacronym. See §4 for further details.

1.3. Further Reading

The following documents and web pages are also available:

- The glossaries-extra documented code

```
texdoc glossaries-extra-code
```

- Gallery: glossaries, glossaries-extra and bib2gls.¹

- FAQs: glossaries, glossaries-extra and bib2gls.²

- Incorporating makeglossaries or makeglossaries-lite or bib2gls into the document build.³

- The bib2gls application.⁴

- The glossaries package.⁵

¹dickimaw-books.com/gallery
²dickimaw-books.com/faq.php
³dickimaw-books.com/latex/buildglossaries/
⁴ctan.org/pkg/bib2gls
⁵ctan.org/pkg/glossaries
2. Package Options

\usepackage[⟨options⟩]{glossaries-extra}

This chapter describes the package options provided by glossaries-extra that are either not defined by the base glossaries package or are modified by glossaries-extra. You can additionally pass the base package options to glossaries-extra. For example, instead of:

\usepackage[nonumberlist]{glossaries}
\usepackage[abbreviations]{glossaries-extra}

you can simply do:

\usepackage[abbreviations,nonumberlist]{glossaries-extra}

It’s better not to load the glossaries package first. Leave glossaries-extra to load it, where possible, to allow for a smoother integration between the two packages.

After glossaries-extra has been loaded, some of the glossaries-extra package options may be changed with:

\glossariesextrasetup{(options)}

where ⟨options⟩ are the same as the relevant package option.

Certain options can only be supplied as package options since the settings need to be known while glossaries-extra is loading.

To change the base glossaries package’s options (that may be changed after the package
has loaded), continue to use:

\setupglossaries{⟨options⟩}

but don’t use any of the options listed here in that command.

2.1. Glossary Lists

nomissingglstext=(boolean)\hspace{1em}default: true; initial: false

If true, this will suppress the warning written to the transcript and the warning text that will appear in the document if the external glossary files haven’t been generated due to an incomplete document build. However, it’s probably simpler just to fix whatever has caused the failure to build the external file or files.

abbreviations

This option has no value and can’t be cancelled. If used, it will automatically create a new glossary with the label abbreviations and redefines \glsxtrabbrvtype to this label. (The file extensions are glg-abr, gls-abr and glo-abr.) In addition, this option defines a short-cut command:

\printabbreviations[⟨options⟩]

which is equivalent to:

\printglossary[type=\glsxtrabbrvtype,⟨options⟩]

If glossaries-extra-bib2gls is also loaded then this option will additionally provide \printunsrtabbrvtype which uses \printunsrtglossary instead.

The title of the new glossary is given by

\abbreviationsname\hspace{1em}initial: Abbreviations

If this command is already defined, it’s left unchanged. Otherwise it’s defined to “Abbreviations” if babel hasn’t been loaded or \acronymname if babel has been loaded. However, if you’re using babel it’s likely you will need to change this. (See §15 for further details.)
2. Package Options

If you don’t use the `abbreviations` package option, the \abbreviationsname command won’t be defined (unless it’s defined by an included language file).

If the `abbreviations` option is used and the `acronym` option provided by the `glossaries` package hasn’t been used, then \acronymtype will be set to \glsxtrabbrvtype so that acronyms defined with \newacronym can be added to the list of abbreviations. If you want acronyms in the main glossary and other abbreviations in the `abbreviations` glossary then you will need to redefine \acronymtype to main:

```latex
\renewcommand*{\acronymtype}{main}
```

Note that there are no analogous options to the `glossaries` package’s `acronymlists` option (or associated commands) as the abbreviation mechanism is handled differently with `glossaries-extra`.

### symbols

This is passed to the base `glossaries` package, but `glossaries-extra` will additionally define:

```latex
\glsxtrnewsymbol[⟨key=value list⟩]{⟨entry-label⟩}{⟨sym⟩}
```

which is equivalent to:

```latex
\newglossaryentry{⟨entry-label⟩}{name={⟨symbol⟩}, sort={⟨entry-label⟩}, type={symbols}, category={symbol}, ⟨options⟩}
```

Note that the `sort` key is set to the ⟨entry-label⟩ not the ⟨symbol⟩ as the symbol will likely contain commands. If this isn’t appropriate, you can override it by using the `sort` key in the optional argument.

This option also sets the `regular` attribute to `true` for the `symbol` category and provides the category post-description hook:

```latex
\glsxtrpostdescsymbol
```

Initial: empty

If `glossaries-extra-bib2gls` is also loaded then this option will additionally provide `\print-`
unsrtsymbols which uses \printunsrtglossary.

\section*{2. Package Options}

This is passed to the base glossaries package but glossaries-extra will additionally define:

\begin{verbatim}
\glsxtrnewnumber[\langle key=value list\rangle]{\langle entry-label\rangle}{\langle num\rangle}
\end{verbatim}

which is equivalent to:

\begin{verbatim}
\newglossaryentry{\langle entry-label\rangle}{name={\langle number\rangle},
  sort={\langle entry-label\rangle},type={numbers},category={number},
  \langle options\rangle}
\end{verbatim}

Note that the sort key is set to the \langle entry-label\rangle. If this isn’t appropriate, you can override it by using the sort key in the optional argument.

This option also sets the regular attribute to true for the number category and provides the category post-description hook:

\begin{verbatim}
\glsxtrpostdescnumber initial: empty
\end{verbatim}

If glossaries-extra-bib2gls is also loaded then this option will additionally provide \printunsrtnumbers which uses \printunsrtglossary.

This is passed to the base glossaries package (which defines \printacronyms and creates a new glossary with the label acronym) but if glossaries-extra-bib2gls is loaded then this option will additionally provide \printunsrtacronyms which uses \printunsrtglossary.

As with the base glossaries package, this option redefines \acronymtype to acronym. Note that this option doesn’t change \glsxtrabbrvtype.

\begin{verbatim}
\acronym=\langle boolean\rangle default: true; initial: false
\end{verbatim}

If acronym=true, this behaves like acronyms. Note that acronym=false won’t work if the base glossaries package was loaded before glossaries-extra.
2. Package Options

This is passed to the base glossaries package but if glossaries-extra-bib2gls is loaded then this option will additionally provide \printunsrtindex which uses \printunsrtglossary.

The base package index option also defines:

\newterm[(key=value list)]{(entry-label)}

This definition is modified by glossaries-extra to additionally set the category to index and sets the description to discard the post-description hook (\nopostdesc) but retain \glsxtrpostdescription so that the category post-description hook can still be applied.

This option also sets the regular attribute to true for the index category and defines an associated category post-description hook:

\glsxtrpostdescindex initial: empty

2.2. Glossary Style Options

\nopostdot=(boolean) default: true; initial: true

This option is provided by glossaries where it simply alters a corresponding conditional that’s used inside \glspostdescription to determine whether or not to insert a full stop.

The postpunc option (see below) redefines \glspostdescription, so the \nopostdot option is modified by glossaries-extra to reset the hook back to its original definition to counteract any use of the postpunc option.

This option will have no effect if the glossary style doesn’t include \glspostdescription. (Use stylemods to ensure that all the predefined styles that show the description have this hook added.)

If you are using bib2gls, you may prefer to use the post-description-dot resource option.

\postdot

This is a shortcut for \nopostdot=false.

\postpunc=(value)
2. Package Options

This option redefines `\gls{postdescription} to display the required punctuation. Note that this means the hook will no longer check for the `nopostdot` conditional.

This option will have no effect if the glossary style doesn’t include `\gls{postdescription}`. (Use `stylemods` to ensure that all the predefined styles that show the description have this hook added.)

The `postpunc` value may either be the required punctuation or one of the following keywords:

- `postpunc=dot`
  This redefines `\gls{postdescription} to use a full stop but also adjusts the space factor. This isn’t exactly the same as `nopostdot=false` since it removes the conditional from `\gls{postdescription}`. If you are using `bib2gls`, you may prefer to use the `post-description-dot` resource option.

- `postpunc=comma`
  This redefines `\gls{postdescription} to a comma.

- `postpunc=none`
  This redefines `\gls{postdescription} to do nothing. This isn’t exactly the same as `nopostdot=true` since it removes the conditional from `\gls{postdescription}`.

- `stylemods=⟨value⟩`  
  `default: default`
  Loads the `glossaries-extra-stylemods` package (see §8.6.5), which patches the styles provided with the base `glossaries` package so that they all use `\gls{postdescription}`. Extra hooks are also provided to make them easier to customize. The value may be one of the following:

- `stylemods=all`
  Loads all styles that are provided by both `glossaries` and `glossaries-extra`.

- `stylemods=default`
2. Package Options

Patches all the predefined styles that have been loaded, without loading any extra styles. This will typically be all the styles that are usually loaded by glossaries (for example, list and long). Package options such as nolist will alter which styles are loaded. In the case of nostyles, no styles will be loaded, so none of them will be patched.

It’s pointless using both stylemods=default and nostyles. Any glossary style packages that are subsequently loaded won’t be patched.

\texttt{stylemods=\langle list\rangle}

For each element \(\langle\text{tag}\rangle\) in \(\langle\text{list}\rangle\), the corresponding package glossary-\(\langle\text{tag}\rangle\) will be loaded. You can use this in combination with nostyles to only load the particular style package or packages that you require (without loading the full set of defaults). For example,

\texttt{\usepackage[nostyles,stylemods={bookindex,longextra}, style=bookindex]{glossaries-extra}}

This prevents the base glossaries package from loading the default set of styles, but loads glossaries-extra-stylemods, glossary-bookindex and glossary-longextra, and then sets the default style to bookindex.

2.3. Loading Other Packages

Some options listed in other sections, such as the stylemods and record options, also load supplementary packages.

\texttt{prefix}

Loads the glossaries-prefix package (if not already loaded).

\texttt{accsupp}

Loads the glossaries-accsupp package (if not already loaded). This option can only be used as a package option (not in \texttt{\glossariesextrasetup}) as glossaries-extra needs to know whether or not to provide accessibility support while it’s loading.
2. Package Options

The glossaries-accsupp package is still experimental and so accessibility features are liable to change.

If you want to define styles that can interface with the accessibility support provided by glossaries-accsupp use the \glsaccess⟨xxx⟩ type of commands instead of \glsentry⟨xxx⟩ (for example, \glsaccesstext instead of \glsentrytext). If glossaries-accsupp hasn’t been loaded those commands are equivalent (for example, \glsaccesstext just does \glsentrytext) but if it has been loaded, then the \glsaccess⟨xxx⟩ commands will add the accessibility information. See §9 for further details.

2.4. Entry Definitions, References and Indexing

\texttt{undefaction=⟨value⟩} \quad \textit{initial: error}

This indicates what to do if an undefined glossary entry is referenced.

Undefined entries can’t be picked up by any commands that iterate over a glossary list. This includes \forglsentries and \glsaddall.

\texttt{undefaction=error}

Produces an error message for undefined glossary entries.

\texttt{undefaction=warn}

Only produces a warning message for undefined glossary entries. The place where the entry has been referenced will be marked with ?? (as with undefined labels or citations). The unknown marker is produced with:

\texttt{\glsxtrundeftag} \quad \textit{initial: ??}

This defaults to two question marks.

Note that \ifglsused will only display ?? in the document text with \texttt{undefaction=warn} if the entry hasn’t been defined, as the underlying boolean variable doesn’t exist and so is neither true nor false. (There will also be a warning in the transcript.) You may prefer to use \texttt{\GlsXtrIfUnusedOrUndefined} instead. See §5.10 for further details.
2. Package Options

If you want to write a custom command that needs to generate a warning or error for an undefined reference, you can use:

\glsxtrundefaction{⟨message⟩}{⟨additional help⟩}

This will produce the unknown marker if used within the document environment. Depending on the undefaction, \glsxtrundefaction will either create an error with the given ⟨message⟩ and ⟨additional help⟩ or will create a warning with the given ⟨message⟩.

\begin{docdef}
⟨value⟩
default: true; initial: false
\end{docdef}

This setting governs where \newglossaryentry can be used (preamble-only or anywhere before the first glossary or anywhere within the document).

Commands like \newabbreviation and \glsxtrnewsymbol that internally use \newglossaryentry are also governed by this option. Other commands, such as \longnewglossaryentry are always preamble-only.

With just the base glossaries package, \newglossaryentry is allowed in the document environment as long as you haven’t used \makenoidxglossaries. There are, however, problems that can occur when entries are defined within the document environment (see the glossaries documentation for further details). To encourage preamble-only use, the glossaries-extra package prohibits the use of \newglossaryentry within the document environment by default, but if you really want this you can use this package option to allow it.

Note that in the case of bib2gls, all entry data is originally defined in bib files. The entry definitions (using commands like \longnewglossaryentry and \newabbreviation) are written to the glstex files that are input in the preamble.

\begin{docdef}
docdef=false
\end{docdef}

Prohibits the use of \newglossaryentry within the document environment. All entries must be defined in the preamble.

\begin{docdef}
docdef=true
\end{docdef}

Permits the use of \newglossaryentry in the document environment provided \makenoidxglossaries hasn’t been used (as per the base glossaries package). This will create a temporary glsdefs file that contains the entry definitions so that they can be available on the next \LaTeX run at the beginning of the document to allow any glossaries in the front matter to display correctly.
2. Package Options

If all your glossaries occur at the end of the document, consider using \texttt{docdef=restricted} instead.

\begin{itemize}
\item \texttt{docdef=restricted}
\end{itemize}

Permits the use of \texttt{\newglossaryentry} in the document environment provided the entry definitions all occur before the first glossary is displayed.

This avoids the need for the \texttt{glsdefs} file. You will still need to take care about any changes made to the category code of characters that are required by the \langle key\rangle=\langle value\rangle mechanism (that is, the comma and equal sign) and any \texttt{makeindex} or \texttt{xindy} special character that occurs in the \texttt{sort} key or label. If any of those characters are made active in the document (for example, through \texttt{babel} shortcuts), then it can cause problems with the entry definition.

This option will allow \texttt{\newglossaryentry} to be used in the document with \texttt{\makenoidx-glossaries}, but note that \texttt{\longnewglossaryentry} remains a preamble-only command.

With this option, if an entry appears in the glossary before it has been defined, an error will occur (or a warning if the \texttt{undefaction=warn} option is used). If you edit your document and either remove an entry or change its label, you may need to delete the document’s temporary files (such as the aux and gls files).

\begin{itemize}
\item \texttt{docdef=atom}
\end{itemize}

This option behaves like \texttt{docdef=restricted} but creates the \texttt{glsdefs} file for atom’s autocomplete support. This file isn’t input by \texttt{glossaries-extra} and so associated problems with the use of this file are avoided, but it allows the autocomplete support to find the labels in the file.

\begin{itemize}
\item A bug fix in \texttt{glossaries} v4.47 has changed the format of the \texttt{glsdefs} file slightly.
\end{itemize}

As with \texttt{docdef=restricted}, entries may be defined in the preamble or anywhere in the document, but they may only be referenced after they have been defined. Entries must be defined before the associated glossary is displayed.

If you need a list of all entry labels for the use of an editor or helper script you may also want to consider the package options \texttt{writeglslabels} and \texttt{writeglslabelnames} provided by the base \texttt{glossaries} package. Note that with these options and with \texttt{docdef=atom}, only the entry labels that are visible to \LaTeX can be saved. So if you are using \texttt{bib2gls} you will only get the labels of the entries that have already been selected by \texttt{bib2gls}. The \texttt{bib} files can be found by parsing the aux file for \texttt{\glsxtr@resource} (listed in the \texttt{src} option or \texttt{\jobname.bib} if \texttt{src} is missing).

\begin{itemize}
\item \texttt{shortcuts=\{\langle value\rangle\}}
\end{itemize}

\texttt{initial: none}
2. Package Options

Unlike the base glossaries package option of the same name, this option isn’t boolean but has multiple values.

Multiple invocations of the `shortcuts` option within the same option list will override each other. Since these options define commands, the action can’t be undone with a later `\glossariesextrasetup`.

```
shortcuts=ac
```

Set the shortcut commands provided by the base glossaries package for acronyms (such as `\ac`) but use the glossaries-extra abbreviation commands, such as `\glsxtrshort` and `\glsxtrlong`, instead of the analogous base commands, such as `\acrshort` and `\acrlong`. See §4.3.2 for further details.

```
shortcuts=abbreviations
```

Sets the abbreviation shortcuts (see §4.3.2). This setting doesn’t switch on the acronym shortcuts provided by the base glossaries package.

```
shortcuts=abbr alias: abbreviations
```

Synonym for `shortcuts=abbreviations`.

```
shortcuts=other
```

Implements the other (non-abbreviation) shortcut commands:

```
\newentry{⟨entry-label⟩}{⟨options⟩}
```

A synonym for `\newglossaryentry`.

```
\newsym[(key=value list)]{⟨entry-label⟩}{⟨sym⟩}
```

A synonym for `\glsxtrnewsymbol` (provided that the `symbols` package option is also used).


A synonym for \glsxtrnewnumber (provided that the numbers package option is also used).

\texttt{shortcuts=acother}

Implements \texttt{shortcuts=ac} and \texttt{shortcuts=other}.

\texttt{shortcuts=abother}

Implements \texttt{shortcuts=abbreviations} and \texttt{shortcuts=other}.

\texttt{shortcuts=all}

Implements \texttt{shortcuts=ac}, \texttt{shortcuts=abbreviations} and \texttt{shortcuts=other}.

\texttt{shortcuts=acronyms}

Sets the shortcuts provided by the base glossaries package for acronyms (such as \texttt{\acs}). See the glossaries package documentation for further details.

Note that the short and long forms (\texttt{\acs} and \texttt{\acl}) don’t use \texttt{\glsxtrshort} and \texttt{\glsxtrlong} but use the original \texttt{\acrshort} and \texttt{\acrlong}, which aren’t compatible with the glossaries-extra abbreviation mechanism. The better option is to use \texttt{shortcuts=ac}.

Don’t use \texttt{shortcuts=acronyms} unless you have reverted \texttt{\newacronym} back to the base glossaries package’s acronym mechanism. See §4.6 for further details.

\texttt{shortcuts=acro}

\textit{alias: acronyms}

Synonym for \texttt{shortcuts=acronyms}.

\texttt{shortcuts=true}

\textit{alias: all}

This setting is provided by the base glossaries package. With glossaries-extra it’s equivalent
2. Package Options

to `shortcuts=all`.

```
shortcuts=false alias: all
```

This setting is provided by the base glossaries package. With glossaries-extra it’s equivalent to `shortcuts=none`.

```
indexcrossrefs=(boolean) default: true; initial: varies
```

This is a boolean option that governs whether or not to automatically index any cross-referenced entries that haven’t been marked as used at the end of the document. These are entries that are identified in one of the cross-referencing fields (`see` and `seealso`) of another used entry as opposed to entries that have the cross-referencing fields set.

Since entries with the `alias` key are intended as synonyms for another term, the target is expected to be indexed so entries with the `alias` key set aren’t affected by this option.

For example:

```
\newglossaryentry{courgette}{name={courgette},
    description={small vegetable marrow}}
\newglossaryentry{marrow}{name={marrow},
    description={long gourd with green skin},
    seealso={courgette}}
```

Suppose that “marrow” is indexed (so that it appears in the glossary with the cross-reference to “courgette”) but if courgette isn’t indexed anywhere in the document (using commands like `\gls` or `\glsadd`) then there will be a broken cross-reference in the marrow location list pointing to courgette, which doesn’t appear in the glossary. With `indexcrossrefs=true`, the courgette entry will be indexed at the end of the document using `\glsadd` with `format=glsxtrunusedformat`, which corresponds to the command `\glsxtrunusedformat`.

Note that this special format `\glsxtrunusedformat` simply does `\unskip` and ignores its argument, which creates a blank location. If any of the cross-referenced entries have been indexed but haven’t been marked as used (for example, with `\glsadd`) then this will cause a spurious comma in the location list. This is a limitation of the way that `makeindex` and `xindy` work as they are general purpose indexing applications which require locations. If you have entries with cross-references, you may want to consider switching to `bib2gls` instead.

Note that `bib2gls` can automatically find dependent entries when it parses the `bib` source file, so the `record` option automatically implements `indexcrossrefs=false`.
2. Package Options

This function is implemented by code added to the end document hook that determines whether or not to use the command \glsxtraddallcrossrefs. This command iterates over all entries in all glossaries, which adds to the overall document build time, especially if you have defined a large number of entries, so this defaults to indexcrossrefs=false, but it will be automatically switched on if you use the see orseealso keys in any entries. See also §5.9.

\texttt{indexcrossrefs=true}

Enables this setting.

\texttt{indexcrossrefs=false}

Disables this setting even if the see orseealso key is present in any entries.

\texttt{autoseeindex=⟨boolean⟩}

\texttt{default: true; initial: true}

This is a boolean option that governs whether or not the see andseealso keys should automatically index the cross-reference when an entry is defined (see §5.9).

With the base glossaries package, the see key was provided as a shortcut for \glssee. For example:

\verbatimverbatim\newglossaryentry{courgette}{name={courgette},
description={small vegetable marrow}}\newglossaryentry{zucchini}{name={zucchini},
description={},
see={courgette}}

is equivalent to:

\verbatimverbatim\newglossaryentry{courgette}{name={courgette},
description={small vegetable marrow}}\newglossaryentry{zucchini}{name={zucchini},
description={}}\glssee{zucchini}{courgette}

This was designed for documents where only entries that are actually used in the document are defined and ensures that the cross-reference is included in the glossary, even though it may not be referenced anywhere in the document. However, it becomes problematic if neither entry is required in the document.
2. Package Options

The glossaries-extra package modifies the action of the \texttt{see} key so that it also saves the value and will only perform the automated \texttt{\glssee} if \texttt{autoseeindex=true}. Similarly for the \texttt{seealso} key.

Note that the \texttt{record} option automatically implements \texttt{autoseeindex=false} as the corresponding action can be implemented with \texttt{bib2gls}'s \texttt{selection} option.

\begin{itemize}
  \item \texttt{autoseeindex=true}

  Enables automatic indexing using \texttt{\glssee} for the \texttt{see} key (as per the base glossaries package) and \texttt{\glsxtrindexseealso} for the \texttt{seealso} key.

  For example, if an entry is defined as

  \begin{verbatim}
  \newglossaryentry{foo}{name={foo},
                     description={},see={bar,baz}}
  \end{verbatim}

  then, with \texttt{autoseeindex=true} and the default \texttt{indexcrossrefs} setting, this is equivalent to

  \begin{verbatim}
  \glssee{foo}{name={foo},description={}}
  \glossariesextrasetup{indexcrossrefs=true}
  \GlsXtrSetField{foo}{see}{bar,baz}
  \end{verbatim}

\end{itemize}

\begin{itemize}
  \item \texttt{autoseeindex=false}

  The value of the \texttt{see} and \texttt{seealso} keys will be stored in their corresponding fields (and can be accessed using commands like \texttt{\glsxtrusesee} and \texttt{\glsxtruseseealso}) but the cross-reference won’t be automatically indexed.

  Note that \texttt{indexcrossrefs} isn’t automatically implemented by the presence of the \texttt{see} key when \texttt{autoseeindex} is false.

For example, if an entry is defined as

\begin{verbatim}
\end{verbatim}
2. Package Options

\newglossaryentry{foo}{name={foo},
description={},see={bar,baz}}

then, with autoseeindex=false and the default indexcrossrefs setting, this is equivalent to

\newglossaryentry{foo}{name={foo},
description={}}
\GlsXtrSetField{foo}{see}{bar,baz}

It’s therefore possible with this option to remove the cross-references from the location lists and set their position within the glossary style.

Another method of preventing the automatic indexing is to define the entries before the external indexing files have been opened with \makeglossaries. Since the appropriate file isn’t open, the information can’t be written to it. This will need the package option seeno-index=ignore to prevent an error occurring.

\textbf{record}=(\textit{value}) \hspace{1cm} \textit{default: only; initial: off}

This setting indicates whether or not bib2gls is required.

This option can only be set in the preamble and can’t be used after \GlsXtrLoadResources or \glsxtrresourcefile.

With the recording setting on (record=only, record=nameref or record=hybrid), any of the commands that would typically index the entry (such as \gls, \glstext or \glsadd) will add a record to the aux file. bib2gls can then read this information to find out which entries have been used. (Remember that commands like \glsentryname don’t index, so any use of these commands won’t add a corresponding record.) See §11 for further details.

The hybrid method additionally performs the standard indexing action that’s required for makeindex or xindy to work, but this can’t be done until bib2gls has created the glstex files that provide the entry definitions. In general, it’s best to avoid the hybrid method.

\begin{itemize}
  \item \textbf{record}=off
\end{itemize}

Indexing is performed as per the base glossaries package using either \makeglossaries or
2. Package Options

\makenoidxglossaries. This setting implements \texttt{undefaction=error}.

\begin{itemize}
  \item \texttt{record=only}
\end{itemize}

Indexing (or recording) is performed by adding \texttt{bib2gls} records in the aux file. Neither \texttt{\makeglossaries} nor \texttt{\makenoidxglossaries} is permitted. Use \texttt{\GlsXtrLoadResources} (or \texttt{\glsxtrresourcefile}) to set up \texttt{bib2gls} resource options. Glossaries should be displayed with the "unsrt" family of commands, such as \texttt{\printunsrtglossary}.

This setting implements \texttt{undefaction=warn}, \texttt{autoseeindex=false}, \texttt{indexcrossrefs=false} \texttt{sort=none}, and automatically loads the supplementary glossaries-extra-bib2gls package. (There should be no need to explicitly load glossaries-extra-bib2gls.)

This option also defines the \texttt{location} and \texttt{group} keys that are set by \texttt{bib2gls} to provide the location list and group information required by the "unsrt" family of commands.

The document build process is (assuming the file is called \texttt{myDoc.tex}):

\begin{itemize}
  \item \texttt{pdflatex myDoc}
  \item \texttt{bib2gls myDoc}
  \item \texttt{pdflatex myDoc}
\end{itemize}

If you want letter groups you will need the \texttt{--group} or \texttt{-g} switch when invoking \texttt{bib2gls}:

\begin{itemize}
  \item \texttt{pdflatex myDoc}
  \item \texttt{bib2gls -g myDoc}
  \item \texttt{pdflatex myDoc}
\end{itemize}

Note that this setting will prevent the \texttt{see} from automatically implementing \texttt{\glssee}. (\texttt{bib2gls} deals with the \texttt{see} field.) You may explicitly use \texttt{\glssee} in the document, but \texttt{bib2gls} will ignore the cross-reference if the \texttt{see} field was already set for that entry.

\begin{itemize}
  \item \texttt{record=nameref}
\end{itemize}

As \texttt{record=only} but uses \texttt{nameref} records, which include the current label information given by \texttt{\@currentlabel} and \texttt{\@currentHref}. This means that the title can be included in the entry locations, if available. This setting also supports location hypertexts that don’t follow a simple \texttt{⟨h-prefix⟩⟨the-counter⟩} format, which can’t be used with other indexing options.

See §11.5.6 for further details of this option.
2. Package Options

This option requires hyperref, otherwise it will fall back on the usual location records.

Note that \@currentHref is always globally updated whenever \refstepcounter is used, but \@currentlabel isn’t. This can cause some undesired side-effects with some settings. Remember also that the indexcounter option increments the associated counter every time an entry is indexed, which affects this option. If the location counter is the default page, only the location number is shown.

\texttt{record=alsoindex} \hspace{1cm} \textit{alias: hybrid}

Deprecated synonym of \texttt{record=hybrid}.

\texttt{record=hybrid}

This is a hybrid setting that uses \texttt{bib2gls} to fetch entry information from \texttt{bib} files, but uses \texttt{makeindex} or \texttt{xindy} to create the glossary files (which are input with \texttt{\printglossary}). Note that this requires a slower and more complicated build process (see below).

This hybrid approach is provided for the rare instances where an existing \texttt{xindy} rule or module is too complicated to convert to a \texttt{bib2gls} rule but the entries need to be fetched from a \texttt{bib} file. There’s no benefit in using this option with \texttt{makeindex}.

This setting does not load glossaries-extra-\texttt{bib2gls}, as \texttt{bib2gls} is only being used to fetch the entry definitions.

Since it’s redundant to make \texttt{bib2gls} also sort and collate locations (in addition to \texttt{xindy} performing these tasks), use the resource options \texttt{sort=none} and \texttt{save-locations=false} for a faster build. Many of the other resource options are likely to be irrelevant.

This setting must be used with \texttt{\makeglossaries} but not with its optional argument. Each glossary should be displayed using \texttt{\printglossary} (or \texttt{\printglossaries} for all of them).

This setting should not be used with \texttt{\makenoidxglossaries}.

You may need to change the transcript file used by \texttt{bib2gls} to avoid a clash with \texttt{xindy}’s transcript file. This can be done with \texttt{bib2gls}’s \texttt{--log-file} or \texttt{-t} option.

The document build process is (assuming the file is called \texttt{myDoc.tex}):
2. Package Options

```latex
pdflatex myDoc
bib2gls myDoc
pdflatex myDoc
makeglossaries myDoc
pdflatex myDoc
```

Note that, in this case, it’s redundant to call `bib2gls` with the `--group` or `-g` switch as `xindy` will insert the group heading information into the corresponding glossary file.

If you want `bib2gls` to form the letter groups then this hybrid method is inappropriate.

```latex
bibglsaux=⟨basename⟩
```

Alternatively, this setting can be implemented with:

```latex\glsxtrsetbibglsaux{⟨basename⟩}
```

This option should only be used once. If used again no new file will be created. If the `⟨basename⟩` value is empty, records will be written to the normal aux file.

A document containing many records can result in a large aux file with information that’s only relevant to `bib2gls`. This option will create a new file called `⟨basename⟩.aux` that will be used to store the records. The file will be skipped by \LaTeX{} but will be picked up by `bib2gls v3.0+` when it inputs the main aux file. Note that this creates an extra write register.

You should still supply the main aux file when you run `bib2gls` as `⟨basename⟩.aux` will only contain the records and not the other information that `bib2gls` requires (such as the resource options).

```latex
equations=⟨boolean⟩
```

This setting will cause the default location counter to automatically switch to `equation` when inside a numbered equation environment, such as `equation` or `align`. The counter can be
explicitly overridden with the \texttt{counter \texttt{\textbackslash glslink}} option.

\begin{verbatim}
\begin{figure}[htbp]
  \centering
  \includegraphics{example-image}
  \caption{Sample \texttt{\textglsfmttext{foobar}} figure}
  \textglsadd{foobar}
\end{figure}
\end{verbatim}

\begin{itemize}
  \item \texttt{floats=} \langle \text{boolean} \rangle \\
    \texttt{default: true; initial: false}
\end{itemize}

This setting will cause the default location counter to automatically switch to the corresponding counter when inside a floating environment, such as figure or table. The counter can be explicitly overridden with the \texttt{counter \texttt{\textbackslash glslink}} option.

Remember that within floats it’s the \texttt{\textbackslash caption} command that actually uses \texttt{\textbackslash refstepcounter}, so indexing before the caption will result in the wrong reference. The commands for use in captions and sections, such as \texttt{\textglsfmttext{} and \texttt{\textglsfmtshort}}, don’t index. (See §5.3). You may want to consider using \texttt{\textglsadd} after the caption (not before). For example:

\begin{verbatim}
\begin{figure}[htbp]
  \centering
  \includegraphics{example-image}
  \caption{Sample \texttt{\textglsfmttext{foobar}} figure}
  \textglsadd{foobar}
\end{figure}
\end{verbatim}

\begin{itemize}
  \item \texttt{indexcounter}
\end{itemize}

This option defines the indexing counter:

\begin{itemize}
  \item \texttt{wrglossary}
\end{itemize}

which is incremented every time an entry is indexed. This option automatically implements \texttt{counter=wrglossary}. This means that each location will link to the relevant part of the page where the indexing occurred (instead of to the top of the page). See the \texttt{\texttt{bib2gls}} documentation for the \texttt{save-index-counter} resource option for more details.

This option is primarily intended for use with \texttt{\texttt{bib2gls}} (v1.4+) and \texttt{hyperref}. It can be used with \texttt{makeindex} or \texttt{xindy}, but it will interfere with the location list collation, so you won’t have ranges and you’ll have duplicate page numbers present.

This option works by incrementing \texttt{wrglossary} with \texttt{\texttt{\textbackslash refstepcounter}} and adding \texttt{\textbackslash label}. This can cause a problem if the indexing occurs in an equation environment as \texttt{amsmath} forbids multiple occurrences of \texttt{\textbackslash label} (resulting in the “Multiple \textbackslash label’s” error). It’s best to change the counter to page or equation when in maths mode with this option. For example:
2. Package Options

2.5. Debugging

\renewcommand{\glslinkpresetkeys}{%\ifmmode \setupglslink{counter=page}\fi} \renewcommand{\glsaddpresetkeys}{%\ifmmode \setupglsadd{counter=page}\fi}

\documentclass{article} 
\usepackage[debug]{glossaries-extra} 
{\begin{document} \gls{example} \end{document}}

\\documentclass{article} \usepackage[debug]{glossaries-extra} \begin{document} \gls{example} \[example]?? \end{document}

\documentclass{article} \usepackage[debug]{glossaries-extra} \begin{document} \gls{example} \[example]?? \end{document}

\documentclass{article} \usepackage[debug]{glossaries-extra} \begin{document} \gls{example} \[example]?? \end{document}

2.5. Debugging

\texttt{debug=(value)} \hspace{1cm} \textit{default: true; initial: false}

Enables debugging information for draft documents. This option is defined by the base glossaries package, but is extended by glossaries-extra to provide additional settings. If no value is provided, \texttt{true} is assumed. The following values are available:

\texttt{debug=false}

This setting is provided by the glossaries package and is the default. This switches off all debugging commands.

\texttt{debug=true}

This setting is provided by the glossaries package and switches on logging information if an entry is indexed before the relevant indexing files have been opened (only applicable with \texttt{makeindex} and \texttt{xindy}). This option is extended by glossaries-extra to also display the label of unknown entries before the ?? marker.

This uses \texttt{\glsshowtargetfonttext} for the annotation, which is provided by glossaries.

\texttt{debug=showaccsupp}
2. Package Options

Provided by glossaries, this setting shows accessibility information (glossaries-accsupp).

\texttt{debug=all}

Implements all debugging options.

\texttt{debug=showwrgloss}

This setting is only available with glossaries-extra. This implements \texttt{debug=true} and shows a marker (\cdot) just before the write operation is performed by indexing commands. With \texttt{record=hybrid} there will be two marks: one for the write operation to the aux file and one for the associated glossary file used by \texttt{makeindex/xindy}. The marker is produced with the command:

\texttt{\glsxtrwrglossmark}

If the \texttt{indexcounter} option has been used, this setting will also mark where the \texttt{wrglossary} counter has been incremented. The marker is produced with the command:

\texttt{\glsxtrwrglosscountermark\{\langle number\rangle\}}

This marker is also inserted before the location in the definition of \texttt{\glsxtrwrglossarylocfmt}.

\texttt{debug=showtargets}

This setting is provided by glossaries and displays the hyperlink target names whenever any glossary-related commands create a hyperlink or hypertarget (for example, \texttt{gls}, \texttt{glstarget} or \texttt{glshypernumber}). The default is to use marginal notes in \LaTeX\'s “outer” mode and inline annotations for “inner” or maths modes. This uses \texttt{\glsshowtargetinner} for inner and maths annotations and \texttt{\glsshowtargetouter} for the outer annotation.

If there are many targets within a single paragraph this can lead to “too many floats”, so glossaries-extra provides a new package option \texttt{showtargets} that can be used to easily switch to inline annotations for outer mode (rather than having to redefine \texttt{\glsshowtargetouter}).

\texttt{showtargets=\langle value\rangle}
2. Package Options

Automatically implements `debug=showtargets` and adjusts the annotations according to the ⟨value⟩. The glossaries-extra package provides supplementary commands to support this option.

\texttt{\glsxtrshowtargetouter{⟨target-name⟩}}

Formats annotations in outer mode. This is initially \texttt{\glsshowtargetouter} to match \texttt{debug=showtargets}.

\texttt{\glsxtrshowtargetinner{⟨target-name⟩}}

Formats annotations in inner mode. This is initially \texttt{\glsshowtargetinner} to match \texttt{debug=showtargets}.

\texttt{\glsshowtargetinnersymleft{⟨name⟩}}

Shows the left annotation and marker. This uses the left symbol marker:

\texttt{\glsxtrshowtargetsymbolleft}

\texttt{\glsshowtargetinnersymright{⟨name⟩}}

Shows the right marker and annotation. This uses the right symbol marker:

\texttt{\glsxtrshowtargetsymbollt}

\texttt{showtargets=left}

A marker is placed to the left of the link/target and a marginal note is used in outer mode.

\texttt{showtargets=right}
2. Package Options

A marker is placed to the right of the link/target and a marginal note is used in outer mode.

\verb|showtargets=innerleft|

A marker and annotation are placed to the left of the link/target in all modes.

\verb|showtargets=innerright|

A marker and annotation are placed to the right of the link/target in all modes.

\verb|showtargets=annoteleft|

Markers are placed on either side of the link/target with the annotation on the left in all modes.

\verb|showtargets=annoteright|

Markers are placed on either side of the link/target with the annotation on the right in all modes.
3. Defining Entries

The base glossaries package provides commands, such as `\newglossaryentry`, to define entries. The glossaries-extra package provides some additional commands, described in §3.1. For abbreviations, see §4. If you use bib2gls, it will write command definitions within the glstex file. See the bib2gls user manual for further information about those commands.

The glossaries user manual warns against using commands such as \gls{} within field values. However, if you really need this, the glossaries-extra package provides \glsxtrp{} (see §5.4). Alternatively, you may want to consider multi (compound) entries instead (see §7).

3.1. Command Definitions

\begin{verbatim}
\longnewglossaryentry{(entry-label)}{(key=value list)}{(description)}
\end{verbatim}

This command is provided by the base glossaries package to cater for entries with descriptions that contain paragraph breaks. (The ⟨key⟩= ⟨value⟩ interface doesn’t support paragraph breaks in the value.) The base package only provides an unstarred version of this command, which automatically inserts \leavevmode\unskip\nopostdesc at the end of the description. The glossaries-extra package replaces this with a single command:

\begin{verbatim}
\glsxtrpostlongdescription
\end{verbatim}

which has the same effect, but can be redefined if required.

The glossaries-extra package provides a starred form:

\begin{verbatim}
\longnewglossaryentry*{(entry-label)}{(key=value list)}{(description)}
\end{verbatim}

This doesn’t insert the hook at the end of the description.

For a general purpose post-description hook, see §8.6.2.

Additionally, the symbols package option provides \glsxtrnewsymbol, and the numbers package option provides \glsxtrnewnumber. See §2.1 for further details.
3. Defining Entries

3.2. Glossary Entry Keys

In addition to the glossary entry keys provided by the base glossaries package (summarised in §II) the glossaries-extra package provides:

\[
\text{category}=(\text{category-label}) \quad \text{initial: general}
\]

Assigns the category label. This should not contain any special or active characters as it’s used to form command names. See §10 for further details.

\[
\text{seealso}=(\text{xr-list})
\]

This key is analogous to the see key but the tag is always given by \seealso. The value (\text{xr-list}) should be a comma-separated list of entry labels. As with the see key, this key automatically indexes the cross-reference by default. The cross-reference will be displayed in the location list using \glsxtrusesalsoformat (see §5.9). Use autoseeindex=false to prevent the automatic indexing. (With bib2gls, adjust the selection criteria.)

With just the base glossaries package, the see key simply performs this automated indexing. With glossaries-extra the value is also saved. Similarly with the seealso key. The value isn’t saved with explicit use of \glsxtrindexseealso or \glssee.

\[
\text{alias}=(\text{xr-label})
\]

This is similar to the see key but the value can only be a single entry label. In addition to automatically indexing the cross-reference, this command will cause the entry with this key to have hyperlinks go to the aliased entry when referenced with commands like \gls. Whenever the entry is indexed with commands like \gls, the indexing will be performed on the target entry (the alias value). See §5.9 for further details.

Any entry that has a see, seealso or alias key set will be added to the glossary by default when using makeindex or xindy. If you don’t want this behaviour, use the autoseeindex=false package option and implement a post-description hook to insert the cross-reference. Alternatively, consider switching to bib2gls.

If you use bib2gls (see §11) then most of the glossary entry keys can be used as analogous fields in the bib file. For example, instead of writing the following code in your \text file:

33
3. Defining Entries

\newglossaryentry{duck}{name={duck},
    description={a waterbird with webbed feet}}
\newabbreviation{svm}{SVM}{support vector machine}

You would write the following in a bib file:

@entry{duck,
    name={duck},
    description={a waterbird with webbed feet}}
@abbreviation{svm,
    short={SVM},
    long={support vector machine},
}

There are, however, some keys that are considered internal fields by bib2gls, in that they are defined as keys by glossaries-extra and may be assigned in the glstex file that’s input by \GlsXtrLoadResources, but they should not be used in the bib files.

For example, the sort key (which is recommended with xindy where the name contains symbols) should not be used in the bib file. Instead, use the sort-field resource option or the system of sort fallbacks to choose the most appropriate field to obtain the sort value (see Gallery: Sorting¹). The group and location keys are also considered internal fields and are only applicable with the “unsrt” family of commands.

The group and location keys are defined by the record=only and record=nameref options and are only applicable with the “unsrt” family of commands.

\begin{verbatim}
\texttt{group=\{<group-label>\}}
\end{verbatim}

This key is used by bib2gls within the glstex file to set the group label. This label is typically a by-product of the sorting method (see §8.6.4). If it is explicitly set without reference to the order it can result in fragmented groups, see Gallery: Logical Glossary Divisions (type vs group vs parent).² The group title can be set with \glsxtrsetgrouptitle. You will need to invoke bib2gls with the \texttt{--group} (or \texttt{-g}) switch to ensure that this key is set, when required.

¹dickimaw-books.com/gallery/index.php?label=bib2gls-sorting
²dickimaw-books.com/gallery/index.php?label=logicaldivisions
3. Defining Entries

Letter groups are a consequence of sorting, not the other way around.

\[ \text{location} = \langle \text{location-list} \rangle \]

Used by \texttt{bib2gls} to store the formatted location list. The unformatted internal location list is stored in the \texttt{loclist} field, as with \texttt{\printnoidxglossary}.

With the “unsrt” family of commands, if the \texttt{location} field isn’t set, then it will try the \texttt{loclist} field instead, using the same method as \texttt{\printnoidxglossary} to display the locations. If you don’t want locations with \texttt{bib2gls}, either use \texttt{nonumberlist} or use the \texttt{save-locations=false} resource option.

The base glossaries package provides \texttt{\glsaddkey} and \texttt{\glsaddstoragekey} to allow custom keys to be defined. The glossaries-extra package additionally provides:

\[ \texttt{\glsxtrprovidestoragekey\{\langle \text{key} \rangle\}\{\langle \text{default value} \rangle\}\{\langle \text{no link cs} \rangle\}} \quad \text{modifier: *} \]

This is like \texttt{\glsaddstoragekey} but does nothing if the key has already been defined. As with \texttt{\glsaddstoragekey}, the starred version switches on field expansion for the given key (provided that it hasn’t already been defined).

\[ \texttt{\glsxtrifkeydefined\{\langle \text{key} \rangle\}\{\langle \text{true} \rangle\}\{\langle \text{false} \rangle\}} \]

Tests if the given key has been defined as a glossary entry key.

3.3. Plurals

Some languages, such as English, have a general rule that plurals are formed from the singular with a suffix appended. This isn’t an absolute rule. There are plenty of exceptions (for example, geese, children, churches, elves, fairies, sheep). The glossaries package allows the \texttt{plural} key to be optional when defining entries. In some cases a plural may not make any sense (for example, the term is a symbol) and in some cases the plural may be identical to the singular.

To make life easier for languages where the majority of plurals can simply be formed by appending a suffix to the singular, the glossaries package lets the \texttt{plural} field default to the value of the \texttt{text} field with \texttt{\glspluralsuffix} appended. This command is defined to be just the letter “s”. This means that the majority of terms don’t need to have the \texttt{plural} supplied as well, and you only need to use it for the exceptions.

For languages that don’t have this general rule, the \texttt{plural} field will always need to be supplied, where needed.
There are other plural fields, such as `firstplural`, `longplural` and `shortplural`. Again, if you are using a language that doesn’t have a simple suffix rule, you’ll have to supply the plural forms if you need them (and if a plural makes sense in the context).

If these fields are omitted, the glossaries package follows these rules:

- If `firstplural` is missing, then `\glspluralsuffix` is appended to the `first` field, if that field has been supplied. If the `first` field hasn’t been supplied but the `plural` field has been supplied, then the `firstplural` field defaults to the `plural` field. If the `plural` field hasn’t been supplied, then both the `plural` and `firstplural` fields default to the `text` field (or `name`, if no `text` field) with `\glspluralsuffix` appended.

- If the `longplural` field is missing, then `\glspluralsuffix` is appended to the `long` field, if the `long` field has been supplied.

- If the `shortplural` field is missing then, with the base glossaries acronym mechanism, `\acrpluralsuffix` is appended to the `short` field.

The last case is changed with glossaries-extra. With this extension package, the `shortplural` field defaults to the `short` field with `\abbrvpluralsuffix` appended unless overridden by category attributes. This suffix command is set by the abbreviation styles. This means that every time an abbreviation style is implemented, `\abbrvpluralsuffix` is redefined, see §4.1.2 for further details.

### 3.4. Entry Aliases

An entry can be made an alias of another entry using the `alias` key. The value should be the label of the other term. There’s no check for the other’s existence when the aliased entry is defined. This is to allow the possibility of defining the other entry after the aliased entry. (For example, when used with bib2gls.)

```latex
\glsxtraliashook{⟨entry-label⟩}
```

This hook is implemented when an entry is defined with the `alias` key set. It does nothing by default. The value of the `alias` field can be obtained with `\glsxtralias{⟨entry-label⟩}`.

If an entry `<entry-1>` is made an alias of `<entry-2>` then:

- If the `see` field wasn’t provided when `<entry-1>` was defined, the `alias` key will automatically trigger

```latex
\glssee{⟨entry-1⟩}{⟨entry-2⟩}
```

- If the `hyperref` package has been loaded then `\gls{⟨entry-1⟩}` will link to `<entry-2>`’s target. (Unless the `targeturl` attribute has been set for `<entry-1>`’s category.)
3. Defining Entries

- With `record=off` or `record=hybrid`, the `noindex` setting will automatically be triggered when referencing ⟨entry-1⟩ with commands like \gls or \glstext. This prevents ⟨entry-1⟩ from having a location list (aside from the cross-reference added with \glssee) unless it’s been explicitly indexed with \glsadd or if the indexing has been explicitly set using `noindex=false`. See §5.9.3 for adjusting the indexing hook.

  Note that with `record=only`, the location list for aliased entries is controlled with bib2gls’s settings.

The value of the alias field can be accessed with \glsxtralias (see §5.9.2).

3.5. Setting or Updating Fields

See §5.11 for accessing field values and §5.15 for testing field values.

Modifications to fields only have an effect from that point onwards and may be localised to the current scope. If you are using `docdef=true`, any changes to the field values won’t be saved in the glsdefs file.

Some of these commands are subtly different from each other. For example, \glsfielddef (provided by the base glossaries package), \glsxtrdeffield and \GlsXtrSetField all assign a value to a field, but \glsfielddef requires that both the entry and the field exist (so it can’t be used to set an unknown internal field), \GlsXtrSetField requires that the entry exists (so it can be used to set an internal field that doesn’t have an associated key provided that the entry has been defined), and \glsxtrdeffield doesn’t perform any existence checks (which means that it can be used to assign internal fields before the entry is actually defined).

The commands described in this section don’t require the field to have an associated glossary entry key, so you need to be careful not to misspell the field labels.

Assigning or changing fields using the commands described here won’t alter related fields. For example, if you use the text key but not the plural key when you define an entry with \newglossaryentry, the plural key will automatically be set as well, but if you change the value of the text field after the entry has been defined, the plural field won’t be changed. Particular care is required if the field contributes in some way to the indexing information, as this information is typically initialised when the entry is first defined. This includes the sort and parent keys, which should not be changed after the entry has been defined.

With bib2gls, entries aren’t defined on the first \LaTeX run. This means that commands that test for existence will produce a warning and (within the document environment) the ?? unknown marker. For example:
3. Defining Entries

\documentclass{article}
\usepackage[record]{glossaries-extra}
\GlsXtrLoadResources[src=myentries,selection=all]
\begin{document}
Defining info
\glsxtrdeffield{sample}{info}{some information}.
Defining note
\GlsXtrSetField{sample}{note}{some note}.

Info: \glsxtrusefield{sample}{info}.
Note: \glsxtrusefield{sample}{note}.
\end{document}

On the first \LaTeX run, this produces:

Defining info . Defining note ??.
Info: some information. Note: .

At this point the sample entry hasn’t been defined, so referencing it in \GlsXtrSetField results in a warning and the double question mark ?? unknown marker in the text. The field (note) isn’t saved, so nothing is shown when the field is referenced with \glsxtrusefield. Whereas \glsxtrdeffield does save the field with the label info associated with the label sample, even though the sample entry hasn’t actually been defined. The field can then be later obtained with \glsxtrusefield. Once \texttt{bib2gls} has been run, the sample entry should now have its definition in the \texttt{glstex} file, which is loaded by \GlsXtrLoadResources and the note field can be set.

\glsxtrdeffield{⟨entry-label⟩}{⟨field-label⟩}{⟨value⟩}

This uses \texttt{etoolbox}'s \texttt{csdef} command to locally set the field given by its internal label \texttt{(field-label)} to \texttt{(value)} for the entry identified by \texttt{(entry-label)}. No existence check is performed.

\glsxtrdeffield{⟨entry-label⟩}{⟨field-label⟩}{⟨value⟩}

This is like \glsxtrdeffield but (protected) fully expands the value before assigning it to the field.

\glsxtrapptocsvfield{⟨entry-label⟩}{⟨field-label⟩}{⟨element⟩}

38
This command is designed for fields that should contain comma-separated lists. If the field hasn’t been defined, this behaves like \glsxtrdeffield otherwise it will append a comma followed by \langle element\rangle (unexpanded) to the field value. No existence check is performed. This field can be iterated over using \glsxtrforcsvfield or formatted using \glsxtrfieldformatcsvlist. See §5.13 for further details.

\glsxtrfieldlistadd\{\langle entry-label\rangle\}\{\langle field\rangle\}\{\langle value\rangle\}

Appends the given value to the given entry’s field (identified using the field’s internal label) using etoolbox’s \listcsadd. The field value can later be iterated over using \glsxtrfielddolistloop or \glsxtrfieldforlistloop.

\glsxtrfieldlistgadd\{\langle entry-label\rangle\}\{\langle field\rangle\}\{\langle value\rangle\}

As above but uses \listcsgadd to make a global change.

\glsxtrfieldlisteadd\{\langle entry-label\rangle\}\{\langle field\rangle\}\{\langle value\rangle\}

As above but uses \listcseadd to expand the value.

\glsxtrfieldlistxadd\{\langle entry-label\rangle\}\{\langle field\rangle\}\{\langle value\rangle\}

As above but uses \listcsxadd to make a global change.

\glsxtrsetfieldifexists\{\langle entry-label\rangle\}\{\langle field-label\rangle\}\{\langle code\rangle\}

This is used by the commands \GlsXtrSetField, \gGlsXtrSetField, \xGlsXtrSetField, \eGlsXtrSetField, \GlsXtrLetField, \csGlsXtrLetField and \GlsXtrLetFieldToField to produce an error (or warning with \texttt{undefaction=warn} if the entry doesn’t exist. This can be redefined to add extra checks (for example, to prohibit changing certain fields).

\GlsXtrSetField\{\langle entry-label\rangle\}\{\langle field-label\rangle\}\{\langle value\rangle\}

This uses etoolbox’s \csdef command to locally set the field given by its internal label \langle field-label\rangle to \langle value\rangle for the entry identified by \langle entry-label\rangle.

This command is written to the glstex file by \texttt{bib2gls} to set fields that don’t have a
3. Defining Entries

This is like \texttt{\GlsXtrSetField} but uses a global assignment.

This is like \texttt{\GlsXtrSetField} but (protected) fully expands the value first.

This is like \texttt{\eGlsXtrSetField} but uses a global assignment.

This uses etoolbox’s \texttt{\cslet} command to locally set the field given by its internal label \texttt{⟨field-label⟩} to \texttt{⟨cs⟩} for the entry identified by \texttt{⟨entry-label⟩}.

This uses etoolbox’s \texttt{\csletcs} command to locally set the field given by its internal label \texttt{⟨field-label⟩} to the control sequence given by \texttt{⟨cs-name⟩} for the entry identified by \texttt{⟨entry-label⟩}.

This assigns the field identified by its internal label \texttt{⟨field1-label⟩} for the entry identified by \texttt{⟨entry1-label⟩} to the value of the field identified by \texttt{⟨field2-label⟩} for the entry identified by \texttt{⟨entry2-label⟩}.
4. Abbreviations

The acronym mechanism implemented by the base glossaries package is insufficiently flexible for some documents. The glossaries-extra package provides a completely different mechanism to deal with abbreviations in a more flexible manner. The two methods are incompatible. However, the glossaries-extra package provides predefined styles that emulate the appearance of the styles provided by the base package. If you have previously used just the base glossaries package, consult Table 4.2 on page 62 for the closest matching abbreviation style.

4.1. Defining Abbreviations

Abbreviations are defined using:

\newabbreviation[(options)]{(entry-label)}{(short)}{(long)}

where (entry-label) is the entry’s label (used in commands like \gls), (short) is the short form (the abbreviation) and (long) is the long form (what the abbreviation is short for). The optional argument (options) may be used to set additional keys (as per the options list in \newglossaryentry), such as type or category.

This command internally uses \newglossaryentry and sets the type to \glsxtrabbrv-type and the category to abbreviation. The category (see §10) determines the abbreviation style. The style for a particular category is set using \setabbreviationstyle. If the optional argument is omitted, the abbreviation category is assumed (see §4.5 for further details).

The following example document sets up three different abbreviation styles: long-short-sc for the abbreviation category, long-only-short-only for the custom genus category, and short-nolong for the custom common category. Note that the custom title category doesn’t have an associated style.

\setabbreviationstyle{long-short-sc}
\setabbreviationstyle[genus]{long-only-short-only}
\setabbreviationstyle[common]{short-nolong}
\newabbreviation{xml}{xml}{extensible markup language}
\newabbreviation xml{xml}{extensible markup language}
\newabbreviation[clostridium]{C.}{Clostridium}
\newabbreviation[category=clostridium]{C.}{Clostridium}
\newabbreviation[category=genus]{myristica}{M.}{Myristica}
4. Abbreviations

\begin{document}
First use: \gls{xml}, \gls{clostridium}, \gls{myristica}, \\
\gls{html}, \gls{dr}.

Next use: \gls{xml}, \gls{clostridium}, \gls{myristica}, \\
\gls{html}, \gls{dr}.
\end{document}

Example 1: Multiple abbreviation styles

First use: extensible markup language (XML), Clostridium, Myristica, 
HTML, Doctor (Dr).

Next use: XML, C., M., HTML, Dr.

If the category doesn’t have an associated style, the style for the abbreviation category will be used, as with the dr entry above, which uses the long-short-sc style because no style has been associated with its custom title category.

There are two categories that have an abbreviation style set by default: abbreviation and acronym. These are initialised as follows:

\setabbreviationstyle{long-short}
\setabbreviationstyle[acronym]{short}

This means that abbreviations defined with the default abbreviation category will show the long form followed by the short form in parentheses on first use, and those defined with the category set to acronym will only show the short form (that is, the long form won’t be shown on first use).

To make it easier to migrate a file containing entries defined with \newacronym, the glossaries-extra package redefines \newacronym to do:

\newabbreviation[type=\acronymtype,category=acronym,⟨options⟩]{⟨entry-label⟩}{⟨short⟩}{⟨long⟩}

Note that this sets the category to acronym, which means that any abbreviations defined with \newacronym will use the short style by default. If you want to use a different style, you need to set the abbreviation style for the acronym category. For example, to use the long-short style:
4. Abbreviations

\setabbreviationstyle{acronym}{long-short}

This must be placed before the first instance of \newacronym.

You can’t use \setacronymstyle with glossaries-extra.

If you have defined any acronym styles with \newacronymstyle, you will have to migrate them over to \newabbreviationstyle. However, most of the predefined abbreviation styles are flexible enough to adapt to common abbreviation formats. It is possible to revert \newacronym back to using the base glossaries package’s acronym mechanism (§4.6), but it should generally not be necessary.

Terms defined with \newabbreviation (and \newacronym) can be referenced in the main document text using commands like \gls. (If you want to use shortcut commands like \ac, use the shortcuts=ac package option.) Remember that you can use the \prereset and \preunset options to reset or unset the first use flag (see §5.10). Alternatively, you can use the commands described in §4.3. For headings and captions, see §5.3.2.

Avoid using \glsfirst, \glsfirstplural, \glstext and \glsplural with abbreviations. Many of the abbreviation styles are too complex to work with these commands (particularly the case-changing variants or with the ⟨insert⟩ final optional argument or with innertextformat). Instead, use commands like \gls, \glsxtrshort, \glsxtrlong and \glsxtrfull.

4.1.1. Abbreviation Fields: long and short

The ⟨short⟩ and ⟨long⟩ arguments are internally assigned with the short and long keys (so don’t use those keys in ⟨options⟩), but the short and long values may first be modified by category attributes, such as markwords or markshortwords. As with other entries, avoid nested links (see §5.4). This means avoid using the \gls-like and \glstext-like commands within ⟨short⟩ and ⟨long⟩.

If an abbreviation can be formed by combining other entries, consider using the multi (compound) entry function (see §7).

4.1.2. Abbreviation Fields: longplural and shortplural

The longplural key defaults to ⟨long⟩\glspluralsuffix and the shortplural key defaults to ⟨short⟩\abbrvpluralsuffix. The aposplural attribute will instead set the short-
4. Abbreviations

plural to \textit{\texttt{\short}} \textbackslash abbrvpluralsuffix and the noshortplural attribute will set shortplural to just \textit{\texttt{\short}} (see §10). If these values are not appropriate, you will need to explicitly set the longplural and shortplural keys in \texttt{\langle options \rangle}.

The short plural suffix \texttt{\abbrvpluralsuffix} is redefined by the abbreviation style. Some styles, such as the long-short style, simply redefine \texttt{\abbrvpluralsuffix} to just:

\begin{itemize}
  \item \texttt{\glsxtrabbrvpluralsuffix} \textit{initial:} \texttt{\glspluralsuffix}
\end{itemize}

which is defined to \texttt{\glspluralsuffix}.

Some styles, such as the long-short-sc style, redefine \texttt{\abbrvpluralsuffix} to include code to counteract the formatting of the abbreviation.

\begin{itemize}
  \item If you want to change the default short plural suffix, you need to redefine \texttt{\glsxtrabbrvpluralsuffix} not \texttt{\abbrvpluralsuffix}. If you don’t want the suffix added, then set the noshortplural attribute to true.
\end{itemize}

4.1.3. Abbreviation Fields: name and description

The name key is set according to the abbreviation style. There should not be any need to explicitly set it. Some styles require the description key to be set in \texttt{\langle options \rangle}, but other styles will set the description to the long form.

4.1.4. Abbreviation Fields: type

Abbreviations can be assigned to a particular glossary using the type key in \texttt{\langle options \rangle}. The default for \texttt{\newabbreviation} is:

\begin{itemize}
  \item \texttt{\glsxtrabbrvtype} \textit{initial:} \texttt{\glsdefaulttype}
\end{itemize}

This is initialised to \texttt{\glsdefaulttype} (the default glossary), but the abbreviations package option will redefine it to abbreviations.

The default type for \texttt{\newacronym} is:

\begin{itemize}
  \item \texttt{\acronymtype} \textit{initial:} \texttt{\glsdefaulttype}
\end{itemize}

This is initialised to \texttt{\glsdefaulttype}, but the acronyms package option will redefine it to acronym.
4. Abbreviations

4.1.5. General Hooks

The following are general purpose hooks used within `\newabbreviation`. Note that there are additional hooks that are used by the abbreviation styles (see §4.5.3.1).

\begin{enumerate}
\item \texttt{\glsxtrnewabbrevpresetkeyhook\{\langle options\rangle\}\{\langle label\rangle\}\{\langle short\rangle\}}
\end{enumerate}

This hook is provided for further customisation, if required. It’s implemented before the entry is defined (before the \texttt{shortplural} and \texttt{longplural} keys supplied in \langle options\rangle are parsed). Does nothing by default. The arguments are a legacy throwback to old versions that didn’t have \texttt{\glsxtrorgshort}.

\begin{enumerate}
\item \texttt{\newabbreviationhook}
\end{enumerate}

This hook is performed just before the entry is defined. Does nothing by default.

4.2. Examples: \texttt{makeindex vs bib2gls}

Example document using \texttt{makeindex}:

\begin{verbatim}
\documentclass{article}
\usepackage{glossaries-extra}
\makeglossaries
\newglossaryentry{sample}{name={sample},description={an example}}
\newabbreviation{xml}{XML}{extensible markup language}
\newacronym{nasa}{NASA}{National Aeronautics and Space Administration}
\begin{document}
First use: \texttt{\gls{sample}}, \texttt{\gls{xml}} and \texttt{\gls{nasa}}.
Next use: \texttt{\gls{sample}}, \texttt{\gls{xml}} and \texttt{\gls{nasa}}.
\printglossaries
\end{document}
\end{verbatim}
Example 2: \newabbreviation vs \newacronym vs \newglossaryentry

First use: sample, extensible markup language (XML) and NASA. Next use: sample, XML and NASA.

Glossary

NASA National Aeronautics and Space Administration 1

sample an example 1

XML extensible markup language 1

Note that the long form of NASA isn’t displayed on the first use of \gls{nasa}. This is because the acronym category uses the short style by default.

In the above example, all entries are placed in the main (default) glossary. The package options abbreviations and acronyms can be used to split them off into separate glossaries.

If you use bib2gls, the analogous bib entry types are @abbreviation and @acronym. The above example can be rewritten to use bib2gls:

\documentclass{article}
\begin{filecontents*}{\jobname.bib}
@entry{sample,
   name={sample},
   description={an example}
}
@abbreviation{xml,
   short={XML},
   long={extensible markup language}
}
@acronym{nasa,
   short={NASA},
   long={National Aeronautics and Space Administration}
}
\end{filecontents*}
\usepackage[record]{glossaries-extra}
\GlsXtrLoadResources
\begin{document}
First use: \gls{sample}, \gls{xml} and \gls{nasa}. Next use: \gls{sample}, \gls{xml} and \gls{nasa}.
4. Abbreviations

Example 3: @abbreviation vs @acronym vs @entry

First use: sample, extensible markup language (XML) and NASA. Next use: sample, XML and NASA.

Glossary

NASA National Aeronautics and Space Administration 1
sample an example 1
XML extensible markup language 1

4.3. Referencing (Using) Abbreviations

Since \newabbreviation internally uses \newglossaryentry, you can reference abbreviations with the \gls-like commands as with other entries. Remember that you can use the prerest and preunset options to reset or unset the first use flag (see §5.10).

In general it’s best not to use \glsfirst, \glsfirstplural, \glstext, \glspplural or their case-changing variants as many of the abbreviation styles are too complicated for those commands. If you specifically want the full form, use \gls with prerest or use \glsxtrfull. If you specifically want the short form for a particular instance, use \gls with preunset or use \glsxtrshort. If you only want the long form for a particular instance, use \glsxtrlong.

If you never want the short form with \gls, use one of the “noshort” styles, such as long-noshort. If you never want the long form with \gls, use one of the “nolong” styles, such as short-nolong.

If you need to use abbreviations in headings or captions, use commands like \glsfmtshort and \glsfmtlong (see §5.3.2). Commands like \glsentryname are likely to contain non-expandable content.

Example:
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\makeglossaries
\newabbreviation{svm}{SVM}{support vector machine}
\newabbreviation{html}{HTML}{hypertext markup language}

\begin{document}
\tableofcontents
\section{Introducing the \texttt{\glsfmtlong{svm}}}
First use: \texttt{\gls{svm}}.
Next use: \texttt{\gls{svm}}.

\section{Introducing \texttt{\gls{html}} (incorrect)}
First use (not!): \texttt{\gls{html}}.
Next use: \texttt{\gls{html}}.

\glsreset{html}
\section{Introducing \texttt{\glsxtrshort{html}} (incorrect)}
First use: \texttt{\gls{html}}.
Next use: \texttt{\gls{html}}.

\glsreset{html}
\section{Introducing \texttt{\glsfmtshort{html}}}
First use: \texttt{\gls{html}}.
Next use: \texttt{\gls{html}}.

\printglossaries
\end{document}
4. Abbreviations

Example 4: Referencing an abbreviation (with \hyperref)

Contents

1 Introducing the support vector machine 1
2 Introducing hypertext markup language (HTML) (incorrect) 1
3 Introducing HTML (incorrect) 1
4 Introducing HTML 1

Glossary

1 Introducing the support vector machine 1
2 Introducing HTML (incorrect) 1
3 Introducing HTML (incorrect) 1
4 Introducing HTML 1

Glossary

HTML hypertext markup language 1
SVM support vector machine 1

In the above example, compare the first section heading (which references an abbreviation with \glsfmtlong) with the second section heading (which references an abbreviation with \gls). Note that the first use of the html entry actually occurs in the table of contents, which results in the full form showing in the table of contents, but only the abbreviation is shown.
4. Abbreviations

in the actual section 2 heading. The PDF bookmark shows the entry label (html) not the abbreviation (HTML). There is also a nested link for section 2 in the table of contents. In some PDF viewers (such as Okular), this will lead to section 2 but, in others (such as Evince), it will lead to the HTML entry target in the glossary. Similarly for section 3.

As with the base glossaries package, the unformatted short and long forms can be obtained with \glsentryshort and \glsentrylong or, for the plural forms, \glsentryshortpl and \glsentrylongpl. These are analogous to commands like \glsentrytext and may be used in expandable contexts. The sentence case versions (\Glsentryshort, \Glsentrylong, \Glsentryshort, and \Glsentrylong) are all robust in glossaries v4.49 and lower. As from glossaries v4.50, they can expand in PDF bookmarks, but outside of PDF bookmarks they will expand to a robust internal command.

Don’t use the base glossaries package’s acronym commands, such as \acrshort. These aren’t compatible with \newabbreviation.

Each abbreviation style has a display full form, which is the format produced with the first use of \gls, and an inline full form, which is the format produced by \glsxtrfull. For some styles, such as long-short, the display and inline forms are identical.

The example below demonstrates the difference between the first use of \gls compared with the inline full form for the footnote abbreviation style. The example also uses \glsfirst to demonstrate that it produces an undesirable result with the selected abbreviation style.
4. Abbreviations

\setabbreviationstyle{footnote}
\newabbreviation{nasa}{NASA} {National Aeronautics and Space Administration}

\begin{document}
\gls{nasa}'s space exploration\ldots
\glsxtrfull{nasa}'s space exploration\ldots
\glsfirst{nasa}'s space exploration\ldots
\end{document}

In the above, the first use of \gls puts the long form in the footnote but correctly inserts the final optional argument before the footnote marker. The inline full form (obtained with \glsxtrfull) doesn’t use a footnote, but instead shows the long form in parentheses after the short form. The insert material is correctly placed after the short form. Compare this with the final line, which uses \glsfirst. This shows the long form in the footnote, but the inserted material can’t be shifted before the footnote marker, which results in the strange “NASA\textsuperscript{2’s}”.

The following commands are included in the set of \glstext-like commands. They have the same options as \glstext and don’t change the first use flag. They will index (unless noindex is used), create a hyperlink (if enabled), and use the post-link hook.

\begin{itemize}
\item \glsxtrshort[\texttt{\langle options\rangle}]{\langle entry-label\rangle}{\langle insert\rangle} \quad \textbf{modifiers: } * + \langle alt-mod\rangle
\item \Glsxtrshort[\texttt{\langle options\rangle}]{\langle entry-label\rangle}{\langle insert\rangle} \quad \textbf{modifiers: } * + \langle alt-mod\rangle
\end{itemize}

Displays the short form using the abbreviation style’s formatting.

\phantomsection
\addcontentsline{toc}{section}{4. Abbreviations}
4. Abbreviations

As above, but sentence case version.

\texttt{\GLSxtrashort[\{options\}]{\{entry-label\}}[\{insert\}] modifiers: \texttt{* + \{alt-mod\}}}

As above, but all caps version.

\texttt{\GLSxtrashortpl[\{options\}]{\{entry-label\}}[\{insert\}] modifiers: \texttt{* + \{alt-mod\}}}

Displays the short plural form using the abbreviation style's formatting.

\texttt{\GLSxtrashortpl[\{options\}]{\{entry-label\}}[\{insert\}] modifiers: \texttt{* + \{alt-mod\}}}

As above, but sentence case version.

\texttt{\GLSxtrashortpl[\{options\}]{\{entry-label\}}[\{insert\}] modifiers: \texttt{* + \{alt-mod\}}}

As above, but all caps version.

\texttt{\GLSxtrashortpl[\{options\}]{\{entry-label\}}[\{insert\}] modifiers: \texttt{* + \{alt-mod\}}}

Displays the long form using the abbreviation style’s formatting. As from v1.49, this command simulates first use by defining \texttt{\glsxtrifwasfirstuse} to do its first argument. This is done via the command:

\texttt{\glsxtrsetlongfirstuse{\{entry-label\}}}

which is defined as:

\begin{verbatim}
\newcommand{\glsxtrsetlongfirstuse}[1]{% \\
\let\glsxtrifwasfirstuse\@firstoftwo%
\}
\end{verbatim}

This command takes the entry label as the argument, which is ignored by default.

To restore the original behaviour, redefine this command as follows:
4. Abbreviations

This command is also used by the case-changing and plural variants listed below.

\texttt{\renewcommand{\glsxtrsetlongfirstuse}{\% \letcs{\glsxtrifwasfirstuse}{@secondoftwo}{\%}}}

\texttt{\Glsxtrlong[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle]} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

As above, but sentence case version.

\texttt{\GLSxtrlong[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle]} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

As above, but all caps version.

\texttt{\glsxtrlongpl[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle]} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

Displays the long plural form using the abbreviation style’s formatting.

\texttt{\Glsxtrlongpl[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle]} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

As above, but sentence case version.

\texttt{\GLSxtrlongpl[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle]} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

As above, but all caps version.

\texttt{\glsxtrfull[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle]} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

Displays the inline full form using the abbreviation style’s formatting. Depending on the style, this may not be the same as the text produced with the first use of \texttt{\gls}.

\texttt{\Glsxtrfull[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle]} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle
4. Abbreviations

As above, but sentence case version.

\texttt{\GLSxtrfull[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩}

As above, but all caps version.

\texttt{\glsxtrfullpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩}

Displays the inline full plural form using the abbreviation style’s formatting. Depending on the style, this may not be the same as the text produced with the first use of \glspl.

\texttt{\GLSxtrfullpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩}

As above, but sentence case version.

\texttt{\GLSxtrfullpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩}

As above, but all caps version.

\texttt{\glsxtrsetupfulldefs}

This hook is used within \glsxtrfull, \glsxtrfullpl and the case-changing variations to initialise \glsxtrifwasfirstuse in case it’s required in the post-link hook. The default definition is to simulate first use. Note that changing this can cause unexpected results with abbreviation styles that set the post-link hook, such as short-postlong-user.

\texttt{\glsxtrfullsaveinsert{⟨entry-label⟩}{⟨insert⟩}}

This hook is used within \glsxtrfull, \glsxtrfullpl and the case-changing variations to initialise the \glsinsert placeholder. The default definition is to use \glsxtrsaveinsert. If the insert isn’t saved, it can’t be used within the post-link hook for the \glsxtrfull etc. This affects the behaviour of the “post-hyphen” abbreviation styles, such as long-hyphen-postshort-hyphen.

4.3.1. Prefixes

If you are using the glossaries-prefix package (which can be loaded via the prefix package option), then there are commands similar to \glsxtrshort and \glsxtrlong that insert
the corresponding prefix and separator at the front if the short or long form, if the prefix has been set and is non-empty. In all cases, the separator is `\glsprefixsep`, as with `\pgls`.

These commands require glossaries-prefix.

\glsxtrshort\[⟨options⟩]\{⟨entry-label⟩\}[⟨insert⟩]  modifiers: * + ⟨alt-mod⟩

As `\glsxtrshort` but inserts the `prefix` field and separator, if the `prefix` value is set and non-empty.

\Pglsxtrshort\[⟨options⟩]\{⟨entry-label⟩\}[⟨insert⟩]  modifiers: * + ⟨alt-mod⟩

As `\pglsxtrshort` but sentence case. Note the initial “P” in the command name, which matches `\Pgls` (similarly for the other prefix sentence case commands).

\PGLSxtrshort\[⟨options⟩]\{⟨entry-label⟩\}[⟨insert⟩]  modifiers: * + ⟨alt-mod⟩

As `\pglsxtrshort` but all caps.

\pglsxtrshortpl\[⟨options⟩]\{⟨entry-label⟩\}[⟨insert⟩]  modifiers: * + ⟨alt-mod⟩

As `\glsxtrshortpl` but inserts the `prefixplural` field and separator, if the `prefixplural` value is set and non-empty.

\Pglsxtrshortpl\[⟨options⟩]\{⟨entry-label⟩\}[⟨insert⟩]  modifiers: * + ⟨alt-mod⟩

As `\pglsxtrshortpl` but sentence case.

\PGLSxtrshortpl\[⟨options⟩]\{⟨entry-label⟩\}[⟨insert⟩]  modifiers: * + ⟨alt-mod⟩

As `\pglsxtrshortpl` but all caps.

\pglsxtrlong\[⟨options⟩]\{⟨entry-label⟩\}[⟨insert⟩]  modifiers: * + ⟨alt-mod⟩
4. Abbreviations

As \glsxtrlong but inserts the prefixfirst field and separator, if the prefixfirst value is set and non-empty.

\Pglsxtrlong[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \pglsxtrlong but sentence case.

\PGLSxtrlong[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \pglsxtrlong but all caps.

\pglsxtrlongpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \glxtrlongpl but inserts the prefixfirstplural field and separator, if the prefixfirstplural value is set and non-empty.

\Pglsxtrlongpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \pglsxtrlongpl but sentence case.

\PGLSxtrlongpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \pglsxtrlongpl but all caps.

4.3.2. Abbreviation Shortcut Commands

The abbreviation shortcut commands can be enabled using the shortcuts=abbreviations package option (or shortcuts=abbr or shortcuts=ac). The provided shortcut commands listed in Table 4.1 on the next page. Note that \glxtenablerecordcount will switch the shortcuts that use the \cglx-like commands to the corresponding \rglx-like command.

4.4. Tagging Initials

Initial tagging allows you to highlight the initials that form the abbreviation when the long form is shown in the glossary.

\GlsXtrEnableInitialTagging{⟨categories⟩}{⟨cs⟩} modifier: *
## 4. Abbreviations

### Table 4.1: Abbreviation Shortcut Commands

<table>
<thead>
<tr>
<th>Shortcut</th>
<th>Shortcut</th>
<th>Equivalent Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>\ab</td>
<td>\ac</td>
<td>\cglsl</td>
</tr>
<tr>
<td>\abp</td>
<td>\acp</td>
<td>\cglslpl</td>
</tr>
<tr>
<td>\as</td>
<td>\acs</td>
<td>\glsxtrshort</td>
</tr>
<tr>
<td>\asp</td>
<td>\acsp</td>
<td>\glsxtrshortpl</td>
</tr>
<tr>
<td>\al</td>
<td>\acl</td>
<td>\glsxtrlong</td>
</tr>
<tr>
<td>\alp</td>
<td>\aclp</td>
<td>\glsxtrlongpl</td>
</tr>
<tr>
<td>\af</td>
<td>\acf</td>
<td>\glsxtrfull</td>
</tr>
<tr>
<td>\afp</td>
<td>\acfp</td>
<td>\glsxtrfullpl</td>
</tr>
<tr>
<td>\Ab</td>
<td>\Ac</td>
<td>\cglsl</td>
</tr>
<tr>
<td>\Abp</td>
<td>\Acp</td>
<td>\cglslpl</td>
</tr>
<tr>
<td>\As</td>
<td>\Acs</td>
<td>\Glsxtrshort</td>
</tr>
<tr>
<td>\Asp</td>
<td>\Acspl</td>
<td>\Glsxtrshortpl</td>
</tr>
<tr>
<td>\Al</td>
<td>\Acl</td>
<td>\Glsxtrlong</td>
</tr>
<tr>
<td>\Alp</td>
<td>\Aclp</td>
<td>\Glsxtrlongpl</td>
</tr>
<tr>
<td>\Af</td>
<td>\Acf</td>
<td>\Glsxtrfull</td>
</tr>
<tr>
<td>\Afp</td>
<td>\Acfp</td>
<td>\Glsxtrfullpl</td>
</tr>
<tr>
<td>\AB</td>
<td>\AC</td>
<td>\cGLS</td>
</tr>
<tr>
<td>\ABP</td>
<td>\ACP</td>
<td>\cGLSpl</td>
</tr>
<tr>
<td>\AS</td>
<td>\ACS</td>
<td>\GLSxtrshort</td>
</tr>
<tr>
<td>\ASP</td>
<td>\ACSP</td>
<td>\GLSxtrshortpl</td>
</tr>
<tr>
<td>\AL</td>
<td>\ACL</td>
<td>\GLSxtrlong</td>
</tr>
<tr>
<td>\ALP</td>
<td>\ACLP</td>
<td>\GLSxtrlongpl</td>
</tr>
<tr>
<td>\AF</td>
<td>\ACF</td>
<td>\GLSxtrfull</td>
</tr>
<tr>
<td>\AFP</td>
<td>\ACFP</td>
<td>\GLSxtrfullpl</td>
</tr>
<tr>
<td>\newabbr</td>
<td>\newabbr</td>
<td>\newabbreviation</td>
</tr>
</tbody>
</table>
4. Abbreviations

\GlsXtrEnableInitialTagging must be placed before the abbreviations are defined.

This command (robustly) defines \( cs \) (a control sequence) to accept a single argument, which is the letter (or letters) that needs to be tagged. The normal behaviour of \( cs \) within the document is to simply do its argument, but in the glossary it's activated for those categories that have the \texttt{tagging} attribute set to "true". For those cases it will use:

\[ \glsxtrtagfont{⟨text⟩} \]

This command defaults to \( \underline{⟨text⟩} \) but may be redefined as required.

The control sequence \( cs \) can't already be defined when used with the unstarred version of \GlsXtrEnableInitialTagging for safety reasons. The starred version will overwrite any previous definition of \( cs \). As with redefining any commands, ensure that you don't redefine something important.

The first argument of \GlsXtrEnableInitialTagging is a comma-separated list of category names. The \texttt{tagging} attribute will automatically be set to true for those categories. You can later set this attribute for other categories (see §10) but this must be done before the glossary is displayed.

For example, the following uses initial tagging for both the \texttt{acronym} and \texttt{abbreviation} categories. The custom command \texttt{itag} is defined as the tagging command.

\begin{verbatim}
\makeglossaries
\GlsXtrEnableInitialTagging{acronym,abbreviation}{itag}
\setabbreviationstyle[acronym]{short-nolong-desc}
\newacronym{radar}{a system for detecting the location and speed of ships, aircraft, etc, through the use of radio waves}{description of this term}
\{radar\} % identifying label
\{radar\} % short form
\ntag{ra}dio \ntag{d}etection \ntag{a}nd \ntag{r}anging
\newabbreviation{xml}{e\ntag{x}tensible \ntag{m}arkup \ntag{l}anguage}
\newabbreviation[tne]{other}{tagging \ntag{n}ot \ntag{e}nabled}
\end{verbatim}
4. Abbreviations

Example 6: Tagging abbreviation initials

First use: radar, extensible markup language (XML), tagging not enabled (TNE).
Long form only: radio detection and ranging, extensible markup language, tagging not enabled.

Glossary

radar (radio detection and ranging) a system for detecting the location and speed of ships, aircraft, etc, through the use of radio waves

TNE tagging not enabled

XML extensible markup language

The underlining of the tagged letters only occurs in the glossary and then only for entries with the tagging attribute set.

4.5. Abbreviation Styles

The style for a particular category is set using:

\setabbreviationstyle[(category)]{(style-name)}

If the (category) is omitted, abbreviation is assumed. Remember that \newacronym sets the category to acronym so with \newacronym you need to change the style with:

\setabbreviationstyle[acronym]{(style-name)}
The abbreviation style must be set before the abbreviation with the corresponding category is defined. If you are using bib2gls, the style must be set before \GlsXtr-LoadResources.

The style associated with the abbreviation category will be used if an abbreviation is defined with a category that doesn’t have an associated style.

Once you have defined an abbreviation with a given category, you can’t subsequently change the style for that category. You can’t have more than one style per category. The default style for the abbreviation category is long-short and the default style for the acronym category is short-nolong.

In the example below, the custom latin category doesn’t have an associated abbreviation style, so it uses the style assigned to the abbreviation category, not the acronym category. The only reason that the “radar” abbreviation (defined with \newacronym) uses the style associated with the acronym category is because the default definition of \newacronym sets category={acronym}.

\usepackage[T1]{fontenc}\usepackage{glossaries-extra}\setabbreviationstyle{long-short-sc}\newabbreviation{html}{html}{hypertext markup language}\setabbreviationstyle[acronym]{footnote}\newacronym{radar}{radar}{radio detection and ranging}\newacronym[category={latin}]{ibid}{ibid}{ibidem}\begin{document}\gls{html}, \gls{radar} and \gls{ibid}.

\printunsrtglossaries\end{document}

Example 7: Category without an associated abbreviation style

Glossary

HTML hypertext markup language
radar radio detection and ranging
ibid ibidem

4.5.1. Predefined Abbreviation Styles

There are two types of abbreviation styles: those that treat the abbreviation as a regular entry (so that \gls uses \glsgenentryfmt and is encapsulated with \glsxtrregularfont) and those that don’t treat the abbreviation as a regular entry (so that \gls uses \glsxtr-
4. Abbreviations

genabbrvfmt and is encapsulated with \glsxtrabbreviationfont). See §5.5.5 for further
details of those commands.

The non-regular abbreviation styles allow for more complex formats than the regular
styles.

The regular entry abbreviation styles set the regular attribute to true for the category
assigned to each abbreviation with that style. This means that on first use, \gls uses the
value of the first field and on subsequent use \gls uses the value of the text field (and
analogously for the plural and case-changing versions).

The non-regular abbreviation styles don’t set the regular attribute, unless it has already
been set, in which case it will be changed to false. The first and text fields (and their
plural forms) are set, but they aren’t used by commands like \gls, which instead use for-
matting commands, such as \glsxtrfullformat and \glsxtrsubsequentfmt, which are
defined by the style.

In both cases, the first use of \gls may not match the text produced by \glsfirst (and
likewise for the plural and case-changing versions).

The short and long fields are set as appropriate and may be accessed through commands
like \glsxtrshort and \glsxtrlong. These may appear slightly differently to the way the
short or long form is displayed within \gls, depending on the style.

The sample file sample-abbr-styles.pdf demonstrates all predefined styles described
here.

For the “sc” styles that use \textsc, be careful about your choice of fonts as some
only have limited support. For example, you may not be able to combine bold and
small-caps. If you’re using pdfLaTeX, I recommend that you at least use the fontenc
package with the T1 option or something similar.

The predefined styles have helper commands to make it easier to modify the format. These
are described in §4.5.1.3.

Table 4.2 lists the nearest equivalent glossaries-extra abbreviation styles for the predefined
acronym styles provided by glossaries, but note that the new styles use different formatting
commands.

The example documents used to illustrate the predefined styles in the sub-sections below
are all in the form (document class and options may vary):

\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
4. Abbreviations

Table 4.2.: Base Acronym Styles `\setacronymstyle{\textit{base-style-name}}` Verses New Abbreviation Styles `\setabbreviationstyle[\textit{category}]{\textit{new-style-name}}`

<table>
<thead>
<tr>
<th>Base Style Name</th>
<th>New Style Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>long-sc-short</td>
<td>long-short-sc</td>
</tr>
<tr>
<td>long-sm-short</td>
<td>long-short-sm</td>
</tr>
<tr>
<td>long-sp-short</td>
<td>long-short</td>
</tr>
<tr>
<td></td>
<td>with <code>\renewcommand{\glsxtrfullsep}{\glsabspace}</code></td>
</tr>
<tr>
<td>short-long</td>
<td>short-long</td>
</tr>
<tr>
<td>sc-short-long</td>
<td>short-sc-long</td>
</tr>
<tr>
<td>sm-short-long</td>
<td>short-sm-long</td>
</tr>
<tr>
<td>long-short-desc</td>
<td>long-short-desc</td>
</tr>
<tr>
<td>long-sc-short-desc</td>
<td>long-short-sc-desc</td>
</tr>
<tr>
<td>long-sm-short-desc</td>
<td>long-short-sm-desc</td>
</tr>
<tr>
<td>long-sp-short-desc</td>
<td>long-short-desc</td>
</tr>
<tr>
<td></td>
<td>with <code>\renewcommand{\glsxtrfullsep}{\glsabspace}</code></td>
</tr>
<tr>
<td>short-long-desc</td>
<td>short-long-desc</td>
</tr>
<tr>
<td>sc-short-long-desc</td>
<td>short-sc-long-desc</td>
</tr>
<tr>
<td>sm-short-long-desc</td>
<td>short-sm-long-desc</td>
</tr>
<tr>
<td>dua</td>
<td>long-noshort</td>
</tr>
<tr>
<td>dua-desc</td>
<td>long-noshort-desc</td>
</tr>
<tr>
<td>footnote</td>
<td>short-footnote</td>
</tr>
<tr>
<td>footnote-sc</td>
<td>short-sc-footnote</td>
</tr>
<tr>
<td>footnote-sm</td>
<td>short-sm-footnote</td>
</tr>
<tr>
<td>footnote-desc</td>
<td>short-footnote-desc</td>
</tr>
<tr>
<td>footnote-sc-desc</td>
<td>short-sc-footnote-desc</td>
</tr>
<tr>
<td>footnote-sm-desc</td>
<td>short-sm-footnote-desc</td>
</tr>
</tbody>
</table>
4. Abbreviations

\setabbreviationstyle{⟨style-name⟩}
\newabbreviation[⟨options⟩]{ex}{⟨short⟩}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert].
\printunsrtglossaries
\end{document}

where ⟨style-name⟩ is the name of the abbreviation style, ⟨short⟩ is either “SHRT FM” or (for the small caps examples) “shrt fm”. The styles that require the description or user1 key to be set will include that in ⟨options⟩ otherwise the optional argument of \newabbreviation will be omitted. The examples with a style that requires \textsmaller will load relsize. The “hyphen” styles set the markwords and markshortwords attributes. Note that hyperref is loaded with the colorlinks option, so the hyperlink text will be red.

The naming scheme for abbreviation styles is as follows:

• ⟨field1⟩[-⟨modifier1⟩][-⟨post⟩]⟨field2⟩[-⟨modifier2⟩]⟨-user⟩

This is for the parenthetical styles. The -⟨modifier⟩ parts may be omitted. These styles display ⟨field1⟩ followed by ⟨field2⟩ in parentheses. If ⟨field1⟩ or ⟨field2⟩ starts with “no” then that element is omitted from the display style (no parenthetical part) but is included in the inline style. If post is present then ⟨field2⟩ is placed after the link text using the post-link hook. Note that this will use the singular form of ⟨field2⟩ by default, even if \glspl is used. The corresponding non-post style will use the matching form for ⟨field2⟩.

If the -⟨modifier⟩ part is present and is one of sc, sm or em, then the field has a font changing command applied to it.

The modifier -only indicates that field is only present according to whether or not the entry has been used.

The modifier -hyphen indicates the style will substitute inter-word spaces (that have been marked up with the markwords or markshortwords attributes) will be changed to spaces if the inserted material starts with a hyphen (but not for the set of \glsxtr-short and \glsxtrlong commands).

If the -user part is present, then the value of the field given by \glsxtruserfield (user1), if set, is inserted into the parenthetical material.

Examples:

- long-noshort-sc: ⟨field1⟩ is the long form, the short form is set in small caps but omitted in the display style.

- long-em-short-em: both the long form and the short form are emphasized. The short form is in parentheses.

- long-short-em: the short form is emphasized but not the long form. The short form is in parentheses.
4. Abbreviations

- **long-short-user**: if the `user1` key has been set, this produces the style \( (\text{long}) (\langle \text{short} \rangle, \langle \text{user1} \rangle) \) otherwise it just produces \( (\text{long}) (\langle \text{short} \rangle) \).

- **long-hyphen-postshort-hyphen**: the short form and the inserted material (provided by the final optional argument of commands like \texttt{\gls}) is moved to the post-link hook. The long form is formatted according to \texttt{\glslonghyphenfont} (or \texttt{\glsfirstlonghyphenfont} on first use). The short form is formatted according to \texttt{\glsabbrvhyphenfont} (or \texttt{\glsfirstabbrvhyphenfont} on first use).

- **⟨style⟩-noreg**
  
  Some styles set the \texttt{regular} attribute. In some cases, there’s a version of the style that doesn’t set this attribute. For example, \texttt{long-em-noshort-em} sets the \texttt{regular} attribute. The \texttt{long-em-noshort-em-noreg} style is a minor variation of that style that sets the attribute to \texttt{false}.

  There are a few “noshort” styles, such as \texttt{long-hyphen-noshort-noreg}, where there isn’t a corresponding regular version. This is because the style won’t work properly with the \texttt{regular} attribute set, but the naming scheme is maintained for consistency with the other “noshort” styles.

- **⟨field1⟩-⟨modifier1⟩-[post]footnote**
  
  The display style uses \texttt{⟨field1⟩} followed by a footnote with the other field in it. If \texttt{post} is present then the footnote is placed after the link text using the post-link hook. The inline style does \texttt{⟨field1⟩} followed by the other field in parentheses.

  If \texttt{⟨modifier1⟩} is present, \texttt{⟨field1⟩} has a font-changing command applied to it.

  Examples:
  
  - **short-footnote**: short form in the text with the long form in the footnote.
  
  - **short-sc-postfootnote**: short form in smallcaps with the long form in the footnote outside of the link text.

  Take care with the footnote styles. Remember that there are some situations where \texttt{\footnote} doesn’t work.

- **⟨style⟩-desc**
  
  Like \texttt{⟨style⟩} but the \texttt{description} key must be provided when defining abbreviations with this style.

  Examples:
  
  - **short-long-desc**: like \texttt{short-long} but requires a description.
  
  - **short-em-footnote-desc**: like \texttt{short-em-footnote} but requires a description.
Not all combinations that fit the above syntax are provided. Pre-version 1.04 styles that didn’t fit this naming scheme are either provided with a synonym (where the former name wasn’t ambiguous) or provided with a deprecated synonym (where the former name was confusing).

4.5.1.1. Regular Styles

The following abbreviation styles set the regular attribute to true for all categories that have abbreviations defined with any of these styles. This means that they are treated like ordinary entries and are encapsulated with \glsxtrregularfont not \glsxtrabbreviationfont. The \gls-like commands are formatted according to \glsgenentryfmt.

4.5.1.1.1. Short Styles

These styles only show the short form on both first use and subsequent use. See §4.5.1.3.1 and §4.5.1.3.5 for style commands.

short-nolong

Only the short form is shown on first use of the \gls-like commands. The inline full form uses the same parenthetical style as short-long (\glsxtrshortlongformat). Font variations are available with short-sc-nolong, short-sm-nolong and short-em-nolong.

\setabbreviationstyle{short-nolong}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 8: The short-nolong abbreviation style


Glossary

SHRT FM  long form

The long form is formatted with \glslongdefaultfont for the \glsxtrlong set of commands.

The short form is formatted with \glsfirstabbrvdefaultfont within the full form and...
4. Abbreviations

with `\glsabbrvdefaultfont` for subsequent use and for the `\glsxtrshort` set of commands.

The name is set to the short form (`\glsxtrshortnolongname`) and the description is set to the unencapsulated long form.

<table>
<thead>
<tr>
<th>short</th>
<th>alias: short-nolong</th>
</tr>
</thead>
</table>

A synonym for `short-nolong`.

<table>
<thead>
<tr>
<th>short-nolong-desc</th>
</tr>
</thead>
</table>

As `short-nolong` but the description must be supplied in the optional argument of `\newabbreviation`. Font variations are available with `short-sc-nolong-desc`, `short-sm-nolong-desc` and `short-em-nolong-desc`.

```
\setabbreviationstyle{short-nolong-desc}
\newabbreviation[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl{ex}[-insert].
\printunsrtglossaries
\end{document}
```

Example 9: The `short-nolong-desc` abbreviation style


**Glossary**

**SHRT FM (long form) sample description**

The name is set to the short form followed by the long form in parentheses (`\glsxtrshortdescname`), and the sort is set to just the short form.

<table>
<thead>
<tr>
<th>short-desc</th>
<th>alias: short-nolong-desc</th>
</tr>
</thead>
</table>

66
4. Abbreviations

A synonym for short-nolong-desc.

nolong-short

The same as short-nolong except for the inline full form, which shows the same parenthetical style as long-short (\glsxtrlongshortformat). Font variations are available with nolong-short-sc, nolong-short-sm and nolong-short-em.

\setabbreviationstyle{nolong-short}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}

Example 10: The nolong-short abbreviation style


Glossary

SHRT FM long form

short-sc-nolong

This style is like short-nolong but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).

\setabbreviationstyle{short-sc-nolong}
\newabbreviation{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}
Example 11: The short-sc-nolong abbreviation style

First: \textsc{shrt fm}-insert. Next: \textsc{shrt fm}-insert. Full: \textsc{shrt fm}-insert (long form). First plural: \textsc{shrt fm}s-insert.

Glossary

\begin{description}
\item[\textsc{shrt fm}] long form
\end{description}

\begin{itemize}
\item \texttt{short-sc} \hfill \textit{alias: short-sc-nolong}
\end{itemize}

A synonym for short-sc-nolong.

\begin{itemize}
\item \texttt{short-sc-nolong-desc}
\end{itemize}

This style is like short-nolong-desc but it uses \texttt{glsxtrscsuffix}, \texttt{glsabbrvscfont} and \texttt{glsfirstabbrvscfont} (see §4.5.1.3.9).

\begin{verbatim}
\setabbreviationstyle{short-sc-nolong-desc}
\newabbreviation[description={sample description}]{ex}{shrt fm}{long form}
\begin{document}
First: \textsc{shrt fm}-insert. Next: \textsc{shrt fm}-insert. Full: \textsc{shrt fm}-insert (long form). First plural: \textsc{shrt fm}s-insert.
\printunsrtglossaries
\end{document}
\end{verbatim}

Example 12: The short-sc-nolong-desc abbreviation style

First: \textsc{shrt fm}-insert. Next: \textsc{shrt fm}-insert. Full: \textsc{shrt fm}-insert (long form). First plural: \textsc{shrt fm}s-insert.

Glossary

\begin{description}
\item[\textsc{shrt fm} (long form)] sample description
\end{description}
4. Abbreviations

**short-sc-desc**

A synonym for **short-sc-nolong-desc**.

**nolong-short-sc**

This style is like **nolong-short** but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).

\setabbreviationstyle{nolong-short-sc}
\newabbreviation{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 13: The nolong-short-sc abbreviation style


**Glossary**

SHRT FM long form

**short-sm-nolong**

This style is like **short-nolong** but it uses \glsxtrsmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).

\setabbreviationstyle{short-sm-nolong}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries

69
4. Abbreviations

Example 14: The short-sm-nolong abbreviation style


Glossary

SHRT FM long form

<table>
<thead>
<tr>
<th>short-sm</th>
<th>alias: short-sm-nolong</th>
</tr>
</thead>
</table>

A synonym for short-sm-nolong.

<table>
<thead>
<tr>
<th>short-sm-nolong-desc</th>
</tr>
</thead>
</table>

This style is like short-nolong-desc but it uses \glsxtrmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).

\setabbreviationstyle{short-sm-nolong-desc}
\newabbreviation*[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 15: The short-sm-nolong-desc abbreviation style


Glossary

SHRT FM (long form) sample description
4. Abbreviations

**short-sm-desc**

A synonym for `short-sm-nolong-desc`.

**nolong-short-sm**

This style is like `nolong-short` but it uses \glsxtrmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).

```latex
\setabbreviationstyle{nolong-short-sm}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glsp{ex}[-insert]. \printunsrtglossaries
\end{document}
```

Example 16: The `nolong-short-sm` abbreviation style


**Glossary**

SHRT FM long form

**short-em-nolong**

This style is like `short-nolong` but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).

```latex
\setabbreviationstyle{short-em-nolong}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glsp{ex}[-insert]. \printunsrtglossaries
\end{document}
```
Example 17: The short-em-nolong abbreviation style


Glossary

\textit{SHRT FM} long form

\textit{short-em} \textit{alias: short-em-nolong}

A synonym for \textit{short-em-nolong}.

\textit{short-em-nolong-desc}

This style is like \textit{short-nolong-desc} but it uses \texttt{glsxtremsuffix}, \texttt{glsabbrvemfont} and \texttt{glsfirstabbrvemfont} (see §4.5.1.3.9).

\setabbreviationstyle{short-em-nolong-desc}
\newabbreviation[description={sample description}]
{ex}{SHRT FM}{long form}
\begin{document} First: \texttt{\gls{ex}}[-insert]. Next: \texttt{\gls{ex}}[-insert]. Full: \texttt{\glsxtrfull{ex}}[-insert]. First plural: \texttt{\glspl{prereset}{ex}}[-insert]. \printunsrtglossaries \end{document}

Example 18: The short-em-nolong-desc abbreviation style


Glossary

\textit{SHRT FM} (long form) sample description
4. Abbreviations

A synonym for `short-em-nolong-desc`.

This style is like `nolong-short` but it uses `\glsxtremsuffix`, `\glsabbrvemfont` and `\glsfirstabbrvemfont` (see §4.5.1.3.9).

```latex
\setabbreviationstyle{nolong-short-em}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtfull{ex}[-insert]. First plural: \glspl{ex}[-insert].
\end{document}
```

Example 19: The `nolong-short-em` abbreviation style


Glossary

**SHRT FM** long form

### 4.5.1.1.2 Long Styles

These styles only show the long form on both first use and subsequent use. See §4.5.1.3.1 and §4.5.1.3.6 for style commands.

Only the long form is shown on first use and subsequent use of the `\gls`-like commands (`\glsxtrlongformat`). The inline full form uses the same parenthetical style as `long-short` (`\glsxtrlongshortformat`). Font variations are available with `long-noshort-sc-desc`, `long-noshort-sm-desc` and `long-noshort-em-desc`.
4. Abbreviations

\setabbreviationstyle{long-noshort-desc}
\newabbreviation[description={sample description}]
{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 20: The long-noshort-desc abbreviation style


Glossary

long form sample description

The long form is formatted with \glsfirstlongdefaultfont on first use and \glslongdefaultfont for subsequent use and for the \glsxtrlong set of commands.

The short form is formatted with \glsfirstabbrvdefaultfont within the inline full form and with \glsabbrvdefaultfont for the \glsxtrshort set of commands.

The name is set to the long form (\glsxtrlongnoshortdescname) and the description must be supplied.

long-desc alias: long-noshort-desc

A synonym for long-noshort-desc.

long-noshort

This is like long-noshort-desc but the name is set to the short form (\glsxtrlongnoshortname) and the description is set to the unencapsulated long form.

\setabbreviationstyle{long-noshort}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
4. Abbreviations

Example 21: The long-noshort abbreviation style


Glossary

SHRT FM long form

long alias: long-noshort

A synonym for long-noshort.

long-noshort-sc

This style is like long-noshort but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).

\setabbreviationstyle{long-noshort-sc}
\newabbreviation{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}
Example 22: The long-noshort-sc abbreviation style


Glossary

SHRT FM long form

long-noshort-sc-desc

This style is like long-noshort-desc but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).

\setabbreviationstyle{long-noshort-sc-desc}
\newabbreviation[description={sample description}]{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 23: The long-noshort-sc-desc abbreviation style


Glossary

long form sample description

long-noshort-sm

This style is like long-noshort but it uses \glsxtrsmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).
4. Abbreviations

\setabbreviationstyle{long-noshort-sm}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 24: The long-noshort-sm abbreviation style


Glossary

\textbf{SHRT FM} long form

\textbf{long-noshort-sm-desc}

This style is like \textit{long-noshort-desc} but it uses \glsxtrsmid, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).

\setabbreviationstyle{long-noshort-sm-desc}
\newabbreviation[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}
4. Abbreviations

Example 25: The long-noshort-sm-desc abbreviation style


Glossary

long form sample description

long-noshort-em

This style is like long-noshort but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).

\setabbreviationstyle{long-noshort-em}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 26: The long-noshort-em abbreviation style


Glossary

SHRT FM long form

long-noshort-em-desc

This style is like long-noshort-desc but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).
4. Abbreviations

\setabbreviationstyle{long-noshort-em-desc}
\newabbreviation[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
  Full: \glsxtrfull{ex}[-insert]. First plural: \glsp{ex}[-insert].
  \printunsrtglossaries
\end{document}

Example 27: The long-noshort-em-desc abbreviation style


Glossary

long form  sample description

long-em-noshort-em

This style is like long-noshort but it uses \glsxtremsuffix, \glsabbrvemfont, \glsfirstabbrvemfont, \glslongemfont and \glsfirstlongemfont (see §4.5.1.3.9). This emphasizes both the long and short forms.

\setabbreviationstyle{long-em-noshort-em}
\newabbreviation[ex]{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
  Full: \glsxtrfull{ex}[-insert]. First plural: \glsp{ex}[-insert].
  \printunsrtglossaries
\end{document}
Example 28: The long-em-noshort-em abbreviation style


Glossary

\textit{SHRT FM} \textit{long form}

This style is like \textit{long-noshort-desc} but it uses \texttt{\textbackslash glsxtrsuffix}, \texttt{\textbackslash glsabbrvemfont}, \texttt{\textbackslash glsfirstabbrvemfont}, \texttt{\textbackslash glslongemfont} and \texttt{\textbackslash glsfirstlongemfont} (see §4.5.1.3.9). This emphasizes both the long and short forms.

\begin{verbatim}
\setabbreviationstyle{long-em-noshort-em-desc}
\newabbreviation[description={sample description}]{ex}{\textit{SHRT FM}}{long form}
\begin{document}
First: \texttt{\textbackslash gls\{ex\}[-insert]. Next: \texttt{\textbackslash gls\{ex\}[-insert]. Full: \texttt{\textbackslash glsxtrfull\{ex\}[-insert]. First plural: \texttt{\textbackslash glspl[prereset]\{ex\}[-insert]. \printunsrtglossaries
\end{document}
\end{verbatim}

Example 29: The long-em-noshort-em-desc abbreviation style


Glossary

\textit{long form} \textit{sample description}

4.5.1.2. Non-Regular Styles

The following abbreviation styles will set the \texttt{regular} attribute to \texttt{false} if it has previously been set. If it hasn’t already been set, it’s left unset. Other attributes may also be set, depending on the style.
4. Abbreviations

The non-regular styles are too complicated to use \gls{genentryfmt} as the display style (with the \gls{like} commands). Instead they use \gls{strgenabbrvfmt}. This means that these styles won’t work if you provide your own custom display style (using \def{genentryfmt}) that doesn’t check for the regular attribute.

Avoid using \gls{first}, \gls{firstplural}, \gls{text} and \gls{plural} (or their case-changing variants) with these styles. There are also some styles that can be problematic with \GLS{name}.

4.5.1.2.1. Long (Short) Styles

These styles show the long form followed by the short form in parentheses on first use. On subsequent use only the short form is shown. See §4.5.1.3.1 and §4.5.1.3.2 for style commands.

\begin{document}
\setabbreviationstyle{long-short}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \gls{full}[prereset]{ex}[-insert]. First plural: \gls{pl}[prereset]{ex}[-insert]. \print{unsrtglossaries} \end{document}

Example 30: The long-short abbreviation style


Glossary

SHRT FM long form

The long form is formatted with \glsfirst{longdefaultfont} within the full form and with \glslong{defaultfont} for the \gls{long} set of commands. The short form is formatted with \glsfirst{abbrvdefaultfont} within the full form and
with \glsabbrvdefaultfont for subsequent use and for the \glsxtrshort set of commands.

The name is set to the short form (\glsxtrlongshortname) and the description is set to the unencapsulated long form.

As long-short but the description must be supplied in the optional argument of \newabbreviation. Font variations are available with long-short-sc-desc, long-short-sm-desc, long-short-em-desc and long-em-short-em-desc.

\begin{document}
First: \Gls{ex}[-insert]. Next: \Gls{ex}[-insert].
Full: \Glspl{ex}[-insert]. First plural:
\Printunsrtglossaries
\end{document}

Example 31: The long-short-desc abbreviation style


Glossary

long form (SHRT FM) sample description

The name and sort are set to the long form followed by the short form in parentheses (\glsxtrlongshortdescname and \glsxtrlongshortdescsort).

This style is like long-short but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).
4. Abbreviations

\setabbreviationstyle{long-short-sc}
\newabbreviation{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}

Example 32: The long-short-sc abbreviation style


Glossary

\textbf{SHRT FM} long form

\textbf{long-short-sc-desc}

This style is like long-short-desc but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).

\setabbreviationstyle{long-short-sc-desc}
\newabbreviation[description={sample description}]{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}
Example 33: The long-short-sc-desc abbreviation style


Glossary

long form (SHRT FM) sample description

Example 34: The long-short-sm abbreviation style


Glossary

SHRT FM long form

Example 34: The long-short-sm abbreviation style


Glossary

SHRT FM long form

long-short-sm-desc

This style is like long-short-desc but it uses \glsxtrsmtitle, \glsabbrvstitle and \glsfirstabbrvstitle (see §4.5.1.3.9).
Example 35: The long-short-sm-desc abbreviation style


Glossary

long form (SHRT FM) sample description

This style is like long-short but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).
Example 36: The long-short-em abbreviation style

form-insert (*SHRT FM*). First plural: long forms-insert (*SHRT FMs*).

Glossary

*SHRT FM* long form

long-short-em-desc

This style is like long-short-desc but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).

\setabbreviationstyle{long-short-em-desc}
\newabbreviation[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl{ex}[-insert]. \end{document}

Example 37: The long-short-em-desc abbreviation style

form-insert (*SHRT FM*). First plural: long forms-insert (*SHRT FMs*).

Glossary

long form (*SHRT FM*) sample description

long-em-short-em

This style is like long-short but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont, \glsfirstlongemfont and \glslongemfont (see §4.5.1.3.9). That is,
both the long and short forms are emphasized.
Example 38: The \texttt{long-em-short-em} abbreviation style

First: \textit{long form}-insert (\textit{SHRT FM}). Next: \textit{SHRT FM}-insert. Full: \textit{long form}-insert (\textit{SHRT FM}). First plural: \textit{long forms}-insert (\textit{SHRT FMs}).

\textbf{Glossary}

\textit{SHRT FM} \textit{long form}

This style is like \texttt{long-short-desc} but it uses \texttt{glslongemfont} and \texttt{glslongemfont} (see §4.5.1.3.9). That is, both the long and short forms are emphasized.
4. Abbreviations

Example 39: The long-em-short-em-desc abbreviation style


Glossary

long form (SHRT FM) sample description

4.5.1.2.2. Long (Short, User) Styles

These styles are like the long (short) styles in §4.5.1.2.1 but additional content can be supplied in the field identified by \glsxtruserfield, which will be placed in the parenthetical content on first use (if set). The inline full form is the same as the display full form.

These styles use the commands \glsxtrusersuffix, \glsabbrvuserfont, \glsfirstabbrvuserfont, \glslonguserfont and \glsfirstlonguserfont (except where noted). See §4.5.1.3.1 and §4.5.1.3.3 for style commands.

If you need to change the font, you can redefine the associated commands (listed above). However, since small caps are awkward because the short plural suffix needs to counteract the small caps, small caps versions are provided.

This style is like long-short but it includes the additional content in the parentheses on first use or the inline full form. The description is obtained from \glsuserdescription, which can be redefined to include the additional information, if required.

\setabbreviationstyle{long-short-user}
\newabbreviation[\user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}
Example 40: The long-short-user abbreviation style

Full: long form-insert (SHRT FM, extra info). First plural: long forms-insert (SHRT FMs, extra info).

Glossary

SHRT FM long form

long-short-user-desc

This style is like long-short-user but the description must be supplied. The name is obtained from \glsxtrlongshortuserdescname.

\setabbreviationstyle{long-short-user-desc}
\newabbreviation[description={sample description},
user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 41: The long-short-user-desc abbreviation style

Full: long form-insert (SHRT FM, extra info). First plural: long forms-insert (SHRT FMs, extra info).

Glossary

long form (SHRT FM, extra info) sample description

This style is incompatible with \GLSname.

If you need to use \GLSname with this style, you’ll have to redefine \glsxtrlongshort-
userdescname so that the field name doesn’t include the entry label. For example:

```latex
\newcommand{\glsxtrlongshortuserdescname}{% 
    \protect\glslonguserfont{\the\glslongtok}% \space(\protect\glsabbrvuserfont{\the\glsshorttok})%
}
```

**long-postshort-user**

This style is like `long-short-user` but the parenthetical material is placed in the post-link hook. Note that, unlike `long-short-user`, the plural form isn’t used in the parenthetical material. If you require this, you will need to redefine `\glsxtrpostusershortformat`.

```latex
\setabbreviationstyle{long-postshort-user}
\newabbreviation[\user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glspl[prereset]{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}
```

**Example 42:** The `long-postshort-user` abbreviation style


**Glossary**

SHRT FM long form

**long-postshort-user-desc**

This style is like `long-postshort-user` but the description must be supplied. The name is obtained from `\glsxtrlongshortuserdescname`.

---

90
4. Abbreviations

\setabbreviationstyle{long-postshort-user-desc}
\newabbreviation[description={sample description},
user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 43: The long-postshort-user-desc abbreviation style

Full: \textit{long form}-insert (SHRT FM, extra info). First plural: \textit{long forms}-insert
(SHRT FM, extra info).

Glossary

long form (SHRT FM, extra info) sample description

This style is incompatible with \texttt{\GLSname}.

If you need to use \texttt{\GLSname} with this style, you’ll have to redefine \texttt{\glsxtrshortlonguserdescname} so that the field name doesn’t include the entry label, as for \texttt{long-short-user-desc}.

long-postshort-sc-user

This style is like \texttt{long-postshort-user} but it uses \texttt{\glsxtrscusersuffix}, \texttt{\glsabbrvscuserfont} and \texttt{\glsfirstabbrvscuserfont}. The name value is obtained from \texttt{\glsxtrlongshortscusername}.

\setabbreviationstyle{long-postshort-sc-user}
\newabbreviation[user1={extra info}]{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}
Example 44: The long-postshort-sc-user abbreviation style


Glossary

SHRT FM long form

long-postshort-sc-user-desc

This style is like long-postshort-sc-user but the description must be supplied. The name is obtained from \glsxtrlongshortuserdescname.

\setabbreviationstyle{long-postshort-sc-user-desc}
\newabbreviation[description={sample description}, user1={extra info}]{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert].
\end{document}

Example 45: The long-postshort-sc-user-desc abbreviation style


Glossary

long form (SHRT FM, extra info) sample description
4. Abbreviations

This style is incompatible with \GLSname.

If you need to use \GLSname with this style, you’ll have to redefine \glssxtrlongshortscuserdescname so that the field name doesn’t include the entry label.

4.5.1.2.3. Short (Long) Styles

These styles show the short form followed by the long form in parentheses on first use. On subsequent use only the short form is shown. See §4.5.1.3.1 and §4.5.1.3.2 for style commands.

\setabbreviationstyle{short-long}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glsxpl{ex}[-insert]. \printunsrtglossaries \end{document}

Example 46: The short-long abbreviation style


Glossary

\begin{tabular}{ll}
SHRT FM & long form
\end{tabular}

The long form is formatted with \glsfirstlongdefaultfont within the full form and with \glslongdefaultfont for the \glssxtrlong set of commands.

The short form is formatted with \glsfirstabbrvdefaultfont within the full form and with \glssabbrvdefaultfont for subsequent use and for the \glssxtrshort set of commands.

The name is set to the short form (\glssxtrlongshortname) and the description is set to
4. Abbreviations

the unencapsulated long form.

**short-long-desc**

As short-long but the description must be supplied in the optional argument of \newabbreviation. Font variations are available with short-sc-long-desc, short-sm-long-desc, short-em-long-desc and short-em-long-em-desc.

\setabbreviationstyle{short-long-desc}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}

Example 47: The short-long-desc abbreviation style


**Glossary**

**SHRT FM (long form)** sample description

The name is set to the short form followed by the long form in parentheses (\glsxtrshortlongdescname), and the sort is set to just the short form (\glsxtrshortlongdescsort).

**short-sc-long**

This style is like short-long but it uses \glsxtrcsvsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).

\setabbreviationstyle{short-sc-long}
\newabbreviation{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural:
4. Abbreviations

Example 48: The short-sc-long abbreviation style


**Glossary**

**SHRT FM** (long form)

---

**short-sc-long-desc**

This style is like short-long-desc but it uses \glsxtrscsuffix, \glsabbrvscfont and \gls-firstabbrvscfont (see §4.5.1.3.9).

\setabbreviationstyle{short-sc-long-desc}
\newabbreviation[description={sample description}]{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 49: The short-sc-long-desc abbreviation style


**Glossary**

**SHRT FM (long form)** sample description
4. Abbreviations

**short-sm-long**

This style is like `short-long` but it uses \glxtrmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).

\setabbreviationstyle{short-sm-long}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}

Example 50: The `short-sm-long` abbreviation style


**Glossary**

SHRT FM long form

**short-sm-long-desc**

This style is like `short-long-desc` but it uses \glxtrmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).

\setabbreviationstyle{short-sm-long-desc}
\newabbreviation[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}
4. Abbreviations

Example 51: The short-sm-long-desc abbreviation style


Glossary

SHRT FM (long form) sample description

short-em-long

This style is like short-long but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).

\setabbreviationstyle{short-em-long}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 52: The short-em-long abbreviation style


Glossary

SHRT FM long form

short-em-long-desc

This style is like short-long-desc but it uses \glsxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).
4. Abbreviations

\setabbreviationstyle{short-em-long-desc}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsextrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 53: The short-em-long-desc abbreviation style

First: *SHRT FM*-insert (long form). Next: *SHRT FM*-insert. Full: 
*SHRT FM*-insert (long form). First plural: *SHRT FMs*-insert (long forms).

Glossary

*SHRT FM* (long form) sample description

short-em-long-em

This style is like short-long but it uses \glsextremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont, \glsfirstlongemfont and \glslongemfont (see §4.5.1.3.9). That is, both the long and short forms are emphasized.

\setabbreviationstyle{short-em-long-em}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsextrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}
4. Abbreviations

Example 54: The short-em-long-em abbreviation style


Glossary

\textit{SHRT FM (long form)}

\texttt{\textbackslash setabbreviationstyle{short-em-long-em-desc}}
\texttt{\newabbreviation[\texttt{description}={sample description}]}{\texttt{ex}}{\texttt{SHRT FM}}{\texttt{long form}}
\texttt{\begin{document} First: \texttt{\texttt{gl\$\texttt{s\{}\texttt{ex}\texttt{\}}[-insert]. Next: \texttt{\texttt{gl\$\texttt{s\{}\texttt{ex}\texttt{\}}[-insert]. Full: \texttt{\texttt{gl\$sxtrfull\{}\texttt{ex}\texttt{\}}[-insert]. First plural: \texttt{\texttt{gl\$sp\{}\texttt{l\{}\texttt{prereset}\texttt{\}}\texttt{ex}\texttt{\}}[-insert]. \texttt{\texttt{\textbackslash printunsrtglossaries}} \texttt{\texttt{\textbackslash end\{}\texttt{document}}} \end{document}}

Example 55: The short-em-long-em-desc abbreviation style


Glossary

\textit{SHRT FM (long form)} sample description

4.5.1.2.4. Short (Long, User) Styles

These styles are like the short (long) styles in §4.5.1.2.3 but additional content can be supplied in the field identified by \texttt{\textbackslash gl\$sxtr\texttt{userfield}}, which will be placed in the parenthetical
content on first use (if set). The inline full form is the same as the display full form.

These styles use the commands \glsxtrusersuffix, \glsabbrvuserfont, \glsfirstabbrvuserfont, \glslonguserfont and \glsfirstlonguserfont (except where noted). See §4.5.1.3.1 and §4.5.1.3.3 for style commands.

**short-long-user**

This style is like short-long but it includes the additional content in the parentheses on first use or the inline full form.

The description is obtained from \glsuserdescription, which can be redefined to include the additional information, if required.

\setabbreviationstyle{short-long-user}
\newabbreviation[user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 56: The short-long-user abbreviation style


**Glossary**

**SHRT FM** long form

**short-long-user-desc**

This style is like short-long-user but the description must be provided. The name is obtained from \glsxtrshortlonguserdescname and the sort value is obtained from \glsxtrshortlongdescsort.
4. Abbreviations

\setabbreviationstyle{short-long-user-desc}
\newabbreviation[description={sample description},
user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glsp{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 57: The short-long-user-desc abbreviation style

Full: SHRT FM-insert (long form, extra info). First plural: SHRT FMs-
insert (long forms, extra info).

Glossary

SHRT FM (long forms, extra info) sample description

This style is incompatible with \GLSname.

If you need to use \GLSname with this style, you’ll have to redefine \glsxtrshortlong-
userdescname so that the field name doesn’t include the entry label. For example:

\newcommand{\glsxtrlongshortuserdescname}{{%
    \protect\glsabbrvuserfont{\the\glsshorttok}%
    \space(\protect\glslonguserfont{\the\glslongtok})%
  }%

This style is like short-long but it includes the additional content in the parentheses on first
use or the inline full form. The parenthetical content is placed in the post-link hook which
can avoid overly long hyperlinks.

The description is obtained from \glsuserdescription, which can be redefined to in-
clude the additional information, if required.
4. Abbreviations

\setabbreviationstyle{short-postlong-user}
\newabbreviation[user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}

Example 58: The short-postlong-user abbreviation style


Glossary

SHRT FM long form

\setabbreviationstyle{short-postlong-user-desc}
\newabbreviation[description={sample description}, user1={extra info}]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries \end{document}

This style is like short-postlong-user but the description must be provided. The name is obtained from \glsxtrshortlonguserdescname. The sort value is the short form.
4. Abbreviations

Example 59: The short-postlong-user-desc abbreviation style


Glossary

SHRT FM (long forms, extra info) sample description

This style is incompatible with \GLSname.

If you need to use \GLSname with this style, you’ll have to redefine \glsxtrshortlong-userdescname so that the field name doesn’t include the entry label, as for short-long-user-desc.

4.5.1.2.5. Hyphen Styles

These styles test if the inserted material start with a hyphen. See §4.5.1.3.1, §4.5.1.3.2 and §4.5.1.3.7 for style commands.

These styles are designed to be used with the markwords attribute and (if the short form has spaces) the markshortwords attribute. If the inserted material starts with a hyphen, the spaces will be replaced with hyphens. This replacement won’t take place if the corresponding attribute wasn’t used to mark the inter-word spaces.

Note that \glsxtrshort and \glsxtrlong (and their plural and case-changing variants) don’t perform the inter-word space substitution. The inline full form is slightly different from the display full form for the “post” styles.

This style is like long-short but checks the inserted material for a leading hyphen. The description is the long form encapsulated with \glslonghyphenfont. The name is obtained from \glsxtrlongshortname, and the sort value is obtained from \glsxtrlonghyphen-shortsort. The inline full form is the same as the display full form.
4. Abbreviations

\setabbreviationstyle{long-hyphen-short-hyphen}
\glssetcategoryattributes{abbreviation}{markwords,markshortwords}\{true\}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 60: The long-hyphen-short-hyphen abbreviation style

long-form-insert (SHRT-FM-insert). First plural: long-forms-insert (SHRT-
FMs-insert).

Glossary

SHRT FM long form

long-hyphen-postshort-hyphen

This style is like long-hyphen-short-hyphen but places the insert and parenthetical material
in the post-link hook for the display full form.

\setabbreviationstyle{long-hyphen-postshort-hyphen}
\glssetcategoryattributes{abbreviation}{markwords,markshortwords}\{true\}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}
4. Abbreviations

Example 61: The long-hyphen-postshort-hyphen abbreviation style


Glossary

SHRT FM long form

Note that the inline full form (\glsxtrfull) doesn’t show the insert in the post-link hook, but instead places it at the end of the link text. This is because only the \gls-like commands (not the \glstext-like commands) set the placeholder \glsinsert to the supplied insert. If you want the insert to show in the parenthetical part of the post-link hook for the inline full form you need to redefine \glsxtrfullsaveinsert:

\renewcommand*{\glsxtrfullsaveinsert}[2]{\def\glsinsert{#2}}

long-hyphen-short-hyphen-desc

This style is like long-hyphen-short-hyphen but the description must be supplied. The name is obtained from \glsxtrlongshortdescname, and the sort value is obtained from \glsxtrlongshortdescsort.

\setabbreviationstyle{long-hyphen-short-hyphen-desc}
\glssetcategoryattributes{abbreviation}{markwords,markshortwords}{true}
\newabbreviation[description={sample description}]
{ex}{\{SHRT FM\}{long form}}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}
Example 62: The long-hyphen-short-hyphen-desc abbreviation style


Glossary

long form (SHRT FM) sample description

long-hyphen-postshort-hyphen-desc

This style is like long-hyphen-short-hyphen-desc but places the insert and parenthetical material in the post-link hook for the display full form.

Note that as with the long-hyphen-postshort-hyphen-desc style, the insert isn’t included in the post-link hook by default for the inline full form. If you want the insert to show in the
post-link hook for the inline full form you need to redefine \glsxtrfullsaveinsert.

\texttt{long-hyphen-noshort-desc-noreg}

This style is like \texttt{long-noshort-desc-noreg} but checks the inserted material for a leading hyphen. The description must be supplied. The name is obtained from \texttt{\glsxtrlongnoshortdescname}, and the sort value is obtained from \texttt{\glsxtrlonghyphennoshortdescsort}.

\begin{verbatim}
\setabbreviationstyle{long-hyphen-noshort-desc-noreg}
\glssetcategoryattributes{abbreviation}{markwords,markshortwords}{true}
\newabbreviation[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document}
First: \texttt{\gls{ex}[-insert]}. Next: \texttt{\gls{ex}[-insert]}. Full: \texttt{\glsxtrfull{ex}[-insert]}. First plural: \texttt{\glsxpl[prereset]{ex}[-insert]}. \\printunsrtglossaries\end{document}
\end{verbatim}

Example 64: The \texttt{long-hyphen-noshort-desc-noreg} abbreviation style


\textbf{Glossary}

\texttt{long form} sample description

\texttt{long-hyphen-noshort-noreg}

This style is like \texttt{long-noshort-noreg} but checks the inserted material for a leading hyphen. The description is set to the unencapsulated long form. The name is obtained from \texttt{\glsxtrlongnoshortname}, and the sort value is obtained from \texttt{\glsxtrlonghyphennoshortsort}.

\begin{verbatim}
\setabbreviationstyle{long-hyphen-noshort-noreg}
\glssetcategoryattributes{abbreviation}{markwords,markshortwords}{true}
\newabbreviation{ex}{SHRT FM}{long form}
\end{verbatim}
4. Abbreviations

Example 65: The long-hyphen-noshort-noreg abbreviation style


Glossary

SHRT FM long form

This style is like short-long but checks the inserted material for a leading hyphen. The description is the long form encapsulated with \glslonghyphenfont. The name is obtained from \glsxtrshortlongname and the sort value is obtained from \glsxtrshorthyphenlongsort.

\setabbreviationstyle{short-hyphen-long-hyphen}
\glssetcategoryattributes{abbreviation}{markwords,markshortwords}{true}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}
4. Abbreviations

Example 66: The short-hyphen-long-hyphen abbreviation style


Glossary

SHRT FM long form

Example 67: The short-hyphen-postlong-hyphen abbreviation style


Glossary

SHRT FM long form

Note that as with the long-hyphen-postshort-hyphen style, the insert isn’t included in the post-link hook by default for the inline full form. If you want the insert to show in the post-link hook for the inline full form you need to redefine \glsxtrfullsaveinsert (as
4. Abbreviations

described above, for the long-hyphen-postshort-hyphen style).

\texttt{short-hyphen-long-hyphen-desc}

This style is like \texttt{short-hyphen-long-hyphen} but the description must be supplied. The name is obtained from \texttt{\glsxtrshortlongdescname}, and the sort is obtained from \texttt{\glsxtrshortlongdescsort}.

\begin{verbatim}
\setabbreviationstyle{short-hyphen-long-hyphen-desc}
\glsssetcategoryattributes{abbreviation}{markwords,markshortwords}
{true}
\newabbreviation[description={sample description}]
{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrglossaries \end{document}
\end{verbatim}

Example 68: The \texttt{short-hyphen-long-hyphen-desc} abbreviation style


Glossary

SHRT FM (long form) sample description

\texttt{short-hyphen-postlong-hyphen-desc}

This style is like \texttt{short-hyphen-long-hyphen-desc} but the insert and parenthetical material are placed in the post-link hook for the display full form.

\begin{verbatim}
\setabbreviationstyle{short-hyphen-postlong-hyphen-desc}
\glsssetcategoryattributes{abbreviation}{markwords,markshortwords}
{true}
\newabbreviation[description={sample description}]
\end{verbatim}
4. Abbreviations

{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 69: The short-hyphen-postlong-hyphen-desc abbreviation
style
SHRT FM-insert (long form). First plural: SHRT-FMs-insert (long-form-
insert).

Glossary

SHRT FM (long form) sample description

Note that as with the long-hyphen-postshort-hyphen style, the insert isn’t included in the
post-link hook by default for the inline full form. If you want the insert to show in the
post-link hook for the inline full form you need to redefine \glsxtrfullsaveinsert (as
described above, for the long-hyphen-postshort-hyphen style).

4.5.1.2.6. Only Styles

These styles only show the long form on first use and only show the short form on subsequent
use. The inline full form is the same as the display full form. See §4.5.1.3.1, §4.5.1.3.2 and
§4.5.1.3.8 for style commands.

The inline full form uses a parenthetical style with the long form followed by the short
form in parentheses.

long-only-short-only

The name is obtained from \glsxtronlyname and the sort value is just the short form. The
description is the long form encapsulated with \glslongonlyfont.

\setabbreviationstyle{long-only-short-only}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert].
Full: \glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries

111
Example 70: The long-only-short-only abbreviation style


Glossary

SHRT FM  long form

Example 71: The long-only-short-only-desc abbreviation style


Glossary

long form  sample description

Example 72: The long-only-short-sc-only abbreviation style


Glossary

long form  sample description

long-only-short-sc-only

This is like long-only-short-only but uses small caps. The name is obtained from \glsxtronlydescname and the sort is obtained from \glsxtronlydescsort.
sconlyname, and it uses \glsabbrvsconlyfont, \glsfirstabbrvsconlyfont and \glsxtrsconlysuffix for the abbreviation fonts and plural suffix.

\setabbreviationstyle{long-only-short-sc-only}
\newabbreviation{ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}

Example 72: The long-only-short-sc-only abbreviation style


Glossary

SHRT FM long form

This is like long-only-short-only-desc but uses small caps. The name is obtained from \glsxtrsconlydescname, and the sort is obtained from \glsxtrsconlydescsort.

\setabbreviationstyle{long-only-short-sc-only-desc}
\newabbreviation[description={sample description}] {ex}{shrt fm}{long form}
\begin{document} First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries \end{document}
4. Abbreviations

Example 73: The long-only-short-sc-only-desc abbreviation style


Glossary

long form sample description

4.5.1.2.7. Footnote Styles

These styles show the short form (\glsxtrshortformat) with the long form as a footnote on first use. On subsequent use only the short form is shown. See §4.5.1.3.1 and §4.5.1.3.4 for style commands.

The inline full form uses the same parenthetical style as short-long (\glsxtrshortlongformat). Font variations are available with short-sc-footnote, short-sm-footnote and short-em-footnote.

The ⟨insert⟩ is placed after the short form on first use and subsequent use of the \gls-like commands.
4. Abbreviations

\setabbreviationstyle{short-footnote}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document}
First:
\gls{ex}[-insert]. Next:
\gls{ex}[-insert]. Full:
\glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 74: The short-footnote abbreviation style

\begin{itemize}
\item First: SHRT FM-insert\footnote{long form}. Next: SHRT FM-insert. Full: SHRT FM-insert (long form). First plural: SHRT FMs-insert\footnote{long form}.
\end{itemize}

Glossary

\begin{verbatim}
SHRT FM  long form
\end{verbatim}

The long form is formatted with \glsfirstlongfootnotefont within the full form and with \glslongfootnotefont for the \glsxtrlong set of commands.

The short form is formatted with \glsfirstabbrvdefaultfont within the full form and with \glsabbrvdefaultfont for subsequent use and for the \glsxtrshort set of commands.

The name is set to the short form (\glsxtrfootnotename) and the description is set to the unencapsulated long form.

This style automatically sets the nohyperfirst attribute to true for the entry’s category.

footnote \textit{alias: short-footnote}

A synonym for short-footnote.

short-footnote-desc

As short-footnote but the description must be supplied in the optional argument of \newabbreviation.
4. Abbreviations

\setabbreviationstyle{short-footnote-desc}
\newabbreviation[description={sample description}]{ex}{SHRT FM}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}


Glossary

SHRT FM (long form) sample description

The name is set to the short form followed by the long form in parentheses (\glsxtrfootnotedescname), and the sort is set to just the short form (\glsxtrfootnotedescsort).

footnote-desc alias: short-footnote-desc

A synonym for short-footnote-desc.

short-postfootnote

Similar to short-footnote but the footnote command is placed in the post-link hook. This avoids the problem of nested hyperlinks caused by the footnote marker hyperlink being inside the link text (which is why the short-footnote style switches on the nohyperfirst attribute). Using the post-link hook makes it possible to check for following punctuation so that the footnote marker can be shifted after the punctuation character.
Example 76: The short-postfootnote abbreviation style


Glossary

SHRT FM long form

---

\setabbreviationstyle{short-postfootnote}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document}
First:
\gls{ex}[-insert]. Next:
\gls{ex}[-insert]. Full:
\glsxtrfull{ex}[-insert]. First plural:
\glspl[prereset]{ex}[-insert].
\printunsrtglossaries
\end{document}

postfootnote \textit{alias: short-postfootnote}

A synonym for short-postfootnote.

short-postfootnote-desc

Similar to short-footnote-desc but the footnote command is placed in the post-link hook as with short-postfootnote.
\setabbreviationstyle{short-postfootnote-desc}
\newabbreviation[description=
{sample description}]
{ex}{SHRT FM}{long form}
\begin{document}
First:
\gls{ex}[-insert]. Next:
\gls{ex}[-insert]. Full:
\glsxtrfull{ex}[-insert]. First plural:
\glspl[prerset]{ex}[-insert].
\printunsrtglossaries
\end{document}

Example 77: The abbreviation style

First: SHRT FM-insert.\footnote{long form} Next: SHRT FM-insert.\footnote{long form}
Full: SHRT FM-insert (long form). First plural: SHRT FMs-insert.\footnote{long form}

Glossary

SHRT FM (long form) sample description

\begin{itemize}
\item[postfootnote-desc]alias: short-postfootnote-desc
A synonym for short-postfootnote-desc.
\item[short-sc-footnote]
This style is like short-footnote but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).
\end{itemize}
Example 78: The short-sc-footnote abbreviation style


Glossary

SHRT FM long form

\textsuperscript{1}long form  
\textsuperscript{2}long forms

This style is like short-footnote-desc but it uses \glsxtrscsuffix, \glsaabbrvscfont and \glsfirstabbbrvscfont (see §4.5.1.3.9).
4. Abbreviations

Example 79: The \textit{short-sc-footnote-desc} abbreviation style

\begin{document}
First: \gls{ex}\{-insert\}. Next: \gls{ex}\{-insert\}. Full: \glsxtrfull{ex}\{-insert\}. First plural: \glspl{ex}\{-insert\}.
\end{document}

\begin{itemize}
  \item \textit{long form}\textsuperscript{1}
  \item \textit{long forms}\textsuperscript{2}
\end{itemize}

\textbf{Glossary}

\textit{SHRT FM} (long form) sample description

\textit{short-sc-postfootnote}

This style is like \textit{short-postfootnote} but it uses \texttt{\glsxtrcsuffix}, \texttt{\glsabbrvcsfont} and \texttt{\glsfirstabbrvcsfont} (see §4.5.1.3.9).
4. Abbreviations

\setabbreviationstyle{short-sc-postfootnote}
\newabbreviation{ex}{shrt fm}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert].
\end{document}

Example 80: The short-sc-postfootnote abbreviation style

\begin{itemize}
\item First: SHRT FM-insert.\textsuperscript{1} Next: SHRT FM-insert. Full: SHRT FM-insert (long form). First plural: SHRT FMs-insert.\textsuperscript{2}
\end{itemize}

Glossary

SHRT FM long form

\begin{itemize}
\item long form
\item long form
\end{itemize}

\textsuperscript{1}long form

\textsuperscript{2}long form

\subsection*{short-sc-postfootnote-desc}

This style is like short-postfootnote-desc but it uses \glsxtrscsuffix, \glsabbrvscfont and \glsfirstabbrvscfont (see §4.5.1.3.9).
4. Abbreviations

Example 81: The short-sc-postfootnote-desc abbreviation style


Glossary

SHRT FM (long form) sample description

¹long form
²long form

short-sm-footnote

This style is like short-footnote but it uses \glsxtrsmfont, \glsabbrvsmfont and \gls-firstabbrvsmfont (see §4.5.1.3.9).
4. Abbreviations

```latex
\setabbreviationstyle{short-sm-footnote}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries\end{document}
```

Example 82: The short-sm-footnote abbreviation style


Glossary

SHRT FM long form

---

short-sm-footnote-desc

This style is like short-footnote-desc but it uses \glsxtrmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).
\setabbreviationstyle{short-sm-footnote-desc}
\newabbreviation[description=sample description]{ex}{SHRT FM}{long form}
\begin{document} First: \gls{ex}-insert. Next: \gls{ex}-insert. Full: \glsxtrfull{ex}-insert. First plural: \glspl{ex}-insert.
\printunsrtglossaries
\end{document}

Example 83: The short-sm-footnote-desc abbreviation style

First: SHRT FM-insert\textsuperscript{1}. Next: SHRT FM-insert. Full: SHRT FM-insert (long form). First plural: SHRT FMs-insert\textsuperscript{2}.

Glossary

SHRT FM (long form) sample description

\textsuperscript{1}long form
\textsuperscript{2}long forms

\begin{itemize}
\item [short-sm-postfootnote]
This style is like short-postfootnote but it uses \glsxtrsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).
\end{itemize}
Example 84: The short-sm-postfootnote abbreviation style


Glossary

SHRT FM long form

This style is like short-postfootnote-desc but it uses \glsxtrmsuffix, \glsabbrvsmfont and \glsfirstabbrvsmfont (see §4.5.1.3.9).
4. Abbreviations

\texttt{\setabbreviationstyle{short-sm-postfootnote-desc}\newabbreviation[description= {sample description}]{ex}{SHRT FM}{long form}\begin{document} First: \gls{ex}\text{-insert}. Next: \gls{ex}\text{-insert}. Full: \glsxtrfull{ex}\text{-insert}. First plural: \glspl{ex}\text{-insert}. \printunsrtglossaries\end{document}

Example 85: The short-sm-postfootnote-desc abbreviation style

First: SHRT FM-insert.\textsuperscript{1} Next: SHRT FM-insert. Full: SHRT FM-insert (long form). First plural: SHRT FMs-insert.\textsuperscript{2}

Glossary

\textit{SHRT FM (long form)} sample description

\begin{flushleft}
\textsuperscript{1}\textit{long form} \\
\textsuperscript{2}\textit{long form}
\end{flushleft}

\texttt{\texttt{short-em-footnote}}

This style is like \texttt{short-footnote} but it uses \texttt{\glsxtremsuffix}, \texttt{\glsabbrvemfont} and \texttt{\glsfirstabbrvemfont} (see §4.5.1.3.9).

\setabbreviationstyle{short-em-footnote}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl[prereset]{ex}[-insert]. \printunsrtglossaries\end{document}

Example 86: The short-em-footnote abbreviation style

First: \textit{SHRT FM-insert\textsuperscript{1}}. Next: \textit{SHRT FM-insert}. Full: \textit{SHRT FM-insert (long form)}. First plural: \textit{SHRT FM-insert\textsuperscript{2}}.

Glossary

\textit{SHRT FM} long form

\begin{itemize}
\item \textit{long form}\textsuperscript{1}
\item \textit{long forms}\textsuperscript{2}
\end{itemize}

\textbf{short-em-footnote-desc}

This style is like \textbf{short-footnote-desc} but it uses \texttt{\glsxremsuffix}, \texttt{\glsabbrvemfont} and \texttt{\glsfirstabbrvemfont} (see §4.5.1.3.9).
Example 87: The short-em-footnote-desc abbreviation style


Glossary

SHRT FM (long form) sample description

¹long form
²long forms

This style is like short-postfootnote but it uses \glsxxtremsuffix, \glsabbrvemfont and \glsfirstabbrvemfont (see §4.5.1.3.9).
4. Abbreviations

\setabbreviationstyle{short-em-postfootnote}
\newabbreviation{ex}{SHRT FM}{long form}
\begin{document}
First: \gls{ex}[-insert]. Next: \gls{ex}[-insert]. Full: \glsxtrfull{ex}[-insert]. First plural: \glspl{ex}[-insert]. \printunsrtglossaries
\end{document}

Example 88: The short-em-postfootnote abbreviation style

First: \textit{SHRT FM}-insert.\textsuperscript{1} Next: \textit{SHRT FM}-insert. Full: \textit{SHRT FM}-insert (long form). First plural: \textit{SHRT FM}-insert.\textsuperscript{2}

Glossary

\textit{SHRT FM} long form

\begin{quote}
\texttt{short-em-postfootnote-desc}

This style is like \texttt{short-postfootnote-desc} but it uses \texttt{glsxtremsuffix}, \texttt{glsabbrvemfont} and \texttt{glsfirstabbrvemfont} (see §4.5.1.3.9).
\end{quote}
4. Abbreviations

Example 89: The
abbreviation style

\begin{document}
First: \textsc{SHRT FM}\[-insert]. Next: \textsc{SHRT FM}\[-insert].
Full: \textsc{SHRT FM}\[-insert] (long form). First plural: \textsc{SHRT FM}\[-insert].
\end{document}

Glossary

\textsc{SHRT FM} (long form) sample description

4.5.1.2.8. Short Styles

These styles only show the short form on both first use and subsequent use. See §4.5.1.3.1 and §4.5.1.3.5 for style commands. They are essentially identical to the corresponding regular style listed in §4.5.1.2.8 except that they change the regular attribute to false.

\texttt{short-nolong-noreg}

This style is a non-regular version of the \texttt{short-nolong} style.

\texttt{short-nolong-desc-noreg}

This style is a non-regular version of the \texttt{short-nolong-desc} style.

\texttt{nolong-short-noreg}

This style is a non-regular version of the \texttt{nolong-short} style.
4. Abbreviations

4.5.1.2.9. Long Styles
These styles only show the long form on both first use and subsequent use. See §4.5.1.3.1 and §4.5.1.3.6 for style commands. They are essentially identical to the corresponding regular style listed in §4.5.1.2.9 except that they change the regular attribute to false.

- **long-noshort-desc-noreg**
  This style is a non-regular version of the long-noshort-desc style.

- **long-noshort-noreg**
  This style is a non-regular version of the long-noshort style.

- **long-em-noshort-em-noreg**
  This style is a non-regular version of the long-em-noshort-em style.

- **long-em-noshort-em-desc-noreg**
  This style is a non-regular version of the long-em-noshort-em-desc style.

4.5.1.3. Formatting Commands and Hooks
These commands are used by the predefined abbreviation styles. These are considered user commands, which you can redefine to customize the style.

4.5.1.3.1. General
These commands apply to all styles.

- **\ifglsxtrinsertinside (true)\else (false)\fi**
  This conditional is used to determine whether the \langle insert\rangle part should go inside or outside of the style’s font formatting commands. The default setting is false.

- **\glsxtrinsertinsidetrue**
4. Abbreviations

Set the insert inside conditional to true.

\glsxtrinsertinsidefalse

Set the insert inside conditional to false.

\glsxtrparen{⟨text⟩}

Used for parenthetical content in the inline full form and also, for some styles, the display full form. Note that this formats the opening and closing parentheses according to the inner formatting, but not the argument, which should already incorporate it. The default definition is:

\newcommand*{\glsxtrparen}[1]{\glsxtrgenentrytextfmt{(}#1\glsxtrgenentrytextfmt{)}}

\glsxtrfullsep{⟨entry-label⟩}

Separator placed before \glsxtrparen. This is a space by default, but it includes the inner formatting. The argument (the entry label) is ignored by default:

\newcommand*{\glsxtrfullsep}[1]{\glsxtrgenentrytextfmt{ }}

You can redefine this to use \glsabsspace if you want to have a non-breakable space if the short form is less than \glsacsmax in width. (You can use \glsabsspace instead, but note that \glsabsspace doesn’t incorporate the inner formatting.)

\glsabbrvdefaultfont{⟨text⟩}

Abbreviation font command used by styles that don’t have specific font markup (for example, long-short but not long-em-short-em). This just does its argument.

\glsfirstabbrvdefaultfont{⟨text⟩}
4. Abbreviations

First use abbreviation font command used by styles that don’t have specific font markup. This is defined to just use \glsabbrvdefaultfont.

\glsxtrdefaultrevert\{⟨text⟩\}

This is the default definition of \glsxtrrevert used by styles that don’t have specific font markup. If you redefine \glsabbrvdefaultfont, you will need to redefine \glsxtrdefaultrevert as applicable.

\glslongdefaultfont\{⟨text⟩\}

Long form font command used by styles that don’t have specific font markup. This just does its argument.

\glsfirstlongdefaultfont\{⟨text⟩\}

First use long form font command used by styles that don’t have specific font markup. This is defined to just use \glslongdefaultfont.

4.5.1.3.2. Parenthetical Styles

These commands apply to the parenthetical styles, such as long–short.

\glsxtrlongshortname

This command should expand to the value of the name key for styles like long–short. The default definition is:

\glsxpabbrvfont\{\the\glsshorttok\}\{\glscategorylabel\}

\glsxtrlongshortdescsort

This command should expand to the sort value used by styles such as long–short–desc. The default definition is:

\expandonce\glsxtrorglong\space (\expandonce\glsxtrorgshort)
4. Abbreviations

Note that this uses the original ⟨long⟩ and ⟨short⟩ values supplied to \newabbreviation.

This command is irrelevant with the “unsrt” family of commands.

\glsxtrlongshortdescname

This command should expand to the name value used by styles such as long–short–desc. The default definition is:

\glsxplongfont{\the\glslongtok}{\glscategorylabel}\% \protect\glsxtrfullsep{\the\glslabeltok}\% \protect\glsxtrparen
{\glsxpabbrvfont{\the\glsshorttok}{\glscategorylabel}}

This essentially expands to ⟨long⟩ ⟨⟨short⟩⟩ but includes the style font changing commands, the inner text formatting, and accessibility support.

\glsxtrshortlongname

This command should expand to the value of the name key for styles like short–long. The default definition is:

\glsxpabbrvfont{\the\glsshorttok}{\glscategorylabel}

\glsxtrshortlongdescsort

This command should expand to the value of the sort key for styles like short–long–desc. The default definition is just \expandonce\glsxtrorgshort.

This command is irrelevant with the “unsrt” family of commands.

\glsxtrshortlongdescname
4. Abbreviations

This command should expand to the value of the name key for styles like short-long-desc.
The default definition is:

\glsxpabbrvfont{\the\glsshorttok}{\glscategorylabel}%
\protect\glsxtrfullsep{\the\glslabeltok}%
\protect\glsxtrparen
\glsxplongfont{\the\glslongtok}{\glscategorylabel}

4.5.1.3.3. User Styles

These commands apply to the “user” styles, such as long-short-user.

\glsxtruserfield

This command should expand to the internal label of the field used to store the additional
information that should be shown in the parenthetical material on first use. The default is
useri, which corresponds to the user1 key.

\glsxtruserparensep

The separator used within the parenthetical content. This defaults to a comma followed by
a space.

\glsxtruserfieldfmt{⟨text⟩}

Used to encapsulate the value of the field given by \glsxtruserfield within \glsxtruserparen
and \GLSxtruserparen. This simply does its argument by default.

The inner formatting with both \glsxtruserparen and \GLSxtruserparen, and the
case-change with the latter, will be included in the argument of \glsxtruserfieldfmt.

For example, to emphasize the user value and separate it with a semi-colon instead of a
comma:

\renewcommand{\glsxtruserparensep}{; }
\renewcommand{\glsxtruserfieldfmt}[1]{\emph{#1}}

\glsabbrvuserfont{⟨text⟩}
4. Abbreviations

Formatting for the “user” short form. This defaults to \glsabbrvdefaultfont.

\glsfirstabbrvuserfont\{⟨text⟩\}

Formatting for the “user” short form shown on first use. This defaults to \glsabbrvuserfont.

\glsxtrusersuffix initial: \glsxtrabbrvpluralsuffix

Short plural suffix used by the “user” styles. This defaults to \glsxtrabbrvpluralsuffix.

\glslonguserfont\{⟨text⟩\}

Formatting for the “user” long form. This defaults to \glsabbrvdefaultfont.

\glsfirstlonguserfont\{⟨text⟩\}

Formatting for the “user” short form shown on first use. This defaults to \glslonguserfont.

\glsabbrvscuserfont\{⟨text⟩\}

Formatting for the “sc-user” short form. This uses \glsabbrvscfont, which in turn uses \textsc to apply a small caps style, so your document font needs to support it.

\textsc uses small capital glyphs for lowercase characters. Uppercase characters show as normal capitals. This means that you need to use lowercase characters in the abbreviation.

\glsfirstabbrvscuserfont\{⟨text⟩\}

Formatting for the “sc-user” short form shown on first use. This defaults to \glsabbrvscuserfont.

\glsxtrscuserrevert\{⟨text⟩\}
4. Abbreviations

Counteracts the effect of \glsabbrvscuserfont. The default is \gsxtrscuserrevert. If you redefine \glsabbrvscuserfont, you will need to redefine \gsxtrscuserrevert as applicable.

\glssxtrscusersuffix

Short plural suffix used by the “sc-user” styles. This defaults to \gsxtrscsuffix.

\glsuserdescription{⟨text⟩}{⟨entry-label⟩}

The description field is set to this, where the ⟨text⟩ argument is the long form, for the “user” styles where the description is set by default. This is defined to ignore its second argument:

\newcommand*{\glsuserdescription}[2]{\glslonguserfont{#1}}

If you want to include the information contained in the field identified by \gsxtruserfield, the second argument provides a way of accessing that field without relying on the \glscurrententrylabel placeholder. For example:

\renewcommand*{\glsuserdescription}[2]{% \glslonguserfont{#1}% \ifglshasfield{\gsxtruserfield}{#2}% {, \glscurrentfieldvalue}% {}% \}

\gsxtruserparen{⟨text⟩}{⟨entry-label⟩}

If the field given by \gsxtruserfield has been set, this essentially does:

\gsxtrfullsep{⟨entry-label⟩}\gsxtrparen{⟨text⟩, ⟨user-value⟩}

otherwise it does:

\gsxtrfullsep{⟨entry-label⟩}\gsxtrparen{⟨text⟩}

It’s a little more complicated than this as the definition includes the inner formatting around the comma and the field value (⟨user-value⟩). The comma separator is given by \gsxtr-
4. Abbreviations

userparensep, and the field value is encapsulated with \glsxtruserfieldfmt (with the inner formatting inside).

If you redefine this command, you will also need to redefine the following one in a similar manner.

\GluXtrparen{⟨text⟩}{⟨entry-label⟩}

As above but the content of the field given by \glsxtruserfield is converted to all caps. Note that simply applying an uppercase command to \glsxtruserparen can fail as it can cause the label to be converted to all caps, which is the reason why a separate command to internally perform the case-change is provided.

\glsxtrlongshortuserdescname

Expands to the value for the name key for styles like long-short-user-desc. The default definition is:

\protect\glslonguserfont{\the\glslongtok}\
\protect\glsxtrparen{\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}

\glsxtrlongshortscusername

Expands to the value for the name key for styles like long-postshort-sc-user styles where the description is automatically set. The default definition is:

\protect\glsabbrvscuserfont{\the\glsshorttok}

\glsxtrlongshortscuserdescname

Expands to the value for the name key for styles like long-postshort-sc-user-desc. The default definition is:
4. Abbreviations

\protect\glslonguserfont{\the\glslongtok}\
\protect\glsxtruserparen
{\protect\glsabbrvscuserfont{\the\glsshorttok}}{\the\glslabeltok}

\glsxtrshortlonguserdescname

Expands to the value for the name key for styles like short-long-user-desc. The default definition is:

\protect\glsabbruserfont{\the\glsshorttok}\
\protect\glsxtruserparen
{\protect\glslonguserfont{\the\glslongpltok}}{\the\glslabeltok}

\glsxtruserlongshortformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt/cs⟩}{⟨short-fmt/cs⟩}

This command is used on the first use of \gls or with \glsxtrfull by styles like long-short-user to format the long form followed by the short form (with optional user information) in parentheses. The default definition is:

\newcommand*{\glsxtruserlongshortformat}[4]{% 
\glsxtrlongformat{#1}{#2}{#3}\
\glsxtrusershortformat{#1}{#4}\
}

\Glsxtruserlongshortformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt/cs⟩}{⟨short-fmt/cs⟩}

As above but for sentence case.

\GLSxtruserlongshortformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt/cs⟩}{⟨short-fmt/cs⟩}
4. Abbreviations

As above but for all caps.

\glsxtruserlongshortplformat\{entry-label\}\{insert\}\{long-fmt/cs\}\{short-fmt/cs\}

This command is used on the first use of \glspl or with \glsxtrfullpl by styles like long-short-user to format the plural long form followed by the plural short form (with optional user information) in parentheses. The default definition is:

\newcommand*{\glsxtruserlongshortplformat}[4]{\glsxtrlongplformat{#1}{#2}{#3}\glsxtrusershortplformat{#1}{#4}}

\GLSxtruserlongshortplformat\{entry-label\}\{insert\}\{long-fmt/cs\}\{short-fmt/cs\}

As above but for sentence case.

\GLSxtruserlongshortplformat\{entry-label\}\{insert\}\{long-fmt/cs\}\{short-fmt/cs\}

As above but for all caps.

\glsxtrusershortlongformat\{entry-label\}\{insert\}\{long-fmt/cs\}\{short-fmt/cs\}

This command is used on the first use of \gls or with \glsxtrfull by styles like short-long-user to format the short form followed by the long form (with optional user information) in parentheses. The default definition is:

\newcommand*{\glsxtrusershortlongformat}[4]{\glsxtrshortformat{#1}{#2}{#3}\glsxtruserlongformat{#1}{#4}}
4. Abbreviations

\Glsxtrusershortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As above but for sentence case.

\GLSxtrusershortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As above but for all caps.

\glsxtrusershortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

This command is used on the first use of \glspl or with \glsxtrfullpl by styles like short-long-user to format the plural short form followed by the plural long form (with optional user information) in parentheses. The default definition is:

\newcommand{\glsxtrusershortlongplformat}[4]{% \glsxtrshortplformat{#1}{#2}{#3}% \glsxtruserlongplformat{#1}{#4}% }

\Glsxtrusershortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As above but for sentence case.

\GLSxtrusershortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As above but for all caps.

\glsxtrusershortformat{⟨entry-label⟩}{⟨fmt-cs⟩}

Used to format the singular short form in parentheses (with \glsxtruserparen) on the first use of \gls or \Gls or with \glsxtrfull or \Glsxtrfull for styles like long-short-user.
4. Abbreviations

The default definition is:

\newcommand*{\glsxtrusershortformat}[2]{
  \glsxtruserparen{\glsxtrshortformat{#1}{#2}{#1}}
}

\glsxtrusershortplformat{⟨entry-label⟩}{⟨fmt-cs⟩}

As \glsxtrusershortformat but for the first use of \glsp or with \glsxtrfull for styles like \texttt{long-short-user}. This has a similar definition to the above but with \glsxtrshortplformat.

\GLSxtrusershortformat{⟨entry-label⟩}{⟨fmt-cs⟩}

As \glsxtrusershortformat but is used with the all caps \texttt{GLS} or \texttt{GLSxtrfull}. This uses \GLSxtruserparen instead of \glsxtruserparen.

\GLSxtrusershortplformat{⟨entry-label⟩}{⟨fmt-cs⟩}

As \glsxtrusershortplformat but is used with the all caps \texttt{GLSp} or \texttt{GLSxtrfullp}. This uses \GLSxtruserparen instead of \glsxtruserparen.

\glsxtrpostusershortformat{⟨entry-label⟩}{⟨fmt-cs⟩}

Used in the post-link hook to format the short form in parentheses for styles like \texttt{long-postshort-user}. The default definition is:

\newcommand*{\glsxtrpostusershortformat}[2]{%
  \glsxtrifallcaps
  \{\GLSxtrusershortformat{#1}{#2}}%
  \{\glsxtrusershortformat{#1}{#2}}%
%
}

Note that this doesn’t check if the plural form was used. If you require this, you will need to redefine this command to include \glsifplural:
4. Abbreviations

\renewcommand*{\glsxtrpostusershortformat}[2]{%
  \glsxifplural
  {\GLSxtrifallcaps
    \{\GLSxtrusershortplformat{#1}{#2}\%
    \{\GLSxtrusershortplformat{#1}{#2}\%
  }
  {\GLSxtrifallcaps
    \{\GLSxtrusershortformat{#1}{#2}\%
    \{\GLSxtrusershortformat{#1}{#2}\%
  }
}
}

\glsxtruserlongformat{⟨entry-label⟩}{⟨fmt-cs⟩}

Used to format the singular long form in parentheses (with \glsxtruserparen) on the first use of \gls or \Gls or with \glsxtrfull for styles like short-long-user. The default definition is:

\newcommand*{\glsxtruserlongformat}[2]{%
  \glsxtruserparen{\glsxtrlongformat{#1}{}{#2}}{#1}%
}

\GLSxtruserlongformat{⟨entry-label⟩}{⟨fmt-cs⟩}

As \glsxtruserlongformat but all caps. This uses \GLSxtruserparen instead of \glsxtruserparen.

\glsxtruserlongplformat{⟨entry-label⟩}{⟨fmt-cs⟩}

As \glsxtruserlongformat but for the first use of \glspl or with \glsxtrfull for styles like short-long-user. This has a similar definition to \glsxtruserlongformat but with
4. Abbreviations

\glsxtrlongplformat.

\GLSxtruserlongplformat\{entry-label\}\{fmt-cs\}

As \glsxtruserlongplformat but all caps. This uses \GLSxtruserparen instead of \glsxtruserparen.

\glsxtrpostuserlongformat\{entry-label\}\{fmt-cs\}

Used in the post-link hook to format the long form in parentheses for styles like short–postlong–user. The default definition is:

\newcommand*{\glsxtrpostuserlongformat}[2]{\%\glsxtrifallcaps{\GLSxtruserlongformat\{#1\}\{#2\}\%\glsxtruserlongformat\{#1\}\{#2\}\%\}

Note that, as with \glsxtrpostusershortformat, this doesn’t check if the plural form was used. If you require this, you will need to redefined this command to include \glsifplural.

4.5.1.3.4. Footnote Styles

These commands are only used by the footnote styles.

\glsxtrfootnotename

This command should expand to the value of the name key. The default definition is:

\glsxpabbrvfont{\the\glsshorttok}{\glscategorylabel}

\glsxtrfootnotedescname

This command should expand to the value of the name key for styles like footnote–desc. The default definition is:
4. Abbreviations

\glsxxpabbrvfont{\the\glsshorttok}{\glscategorylabel}%
\protect\glsxtrfullsep{\the\glslabeltok}%
\protect\glsxtrparen
{\glsxplongfont{\the\glslongtok}{\glscategorylabel}}%

\glsxtrfootnotedescsort

This command should expand to the value of the `sort` key for styles like `footnote-desc`. The default definition is simply `\the\glsshorttok`.

\glsxtrfootnotedescsort

This command is irrelevant with the “unsrt” family of commands.

\glslongfootnotefont{⟨text⟩}

The formatting command used for the long form in the footnote styles. The default is to simply use `\glslongdefaultfont`.

\glsxfirstlongfootnotefont{⟨text⟩}

The formatting command used for the first use long form in the footnote styles. The default is to simply use `\glslongfootnotefont`.

\glsxtrabbrvfootnote{⟨entry-label⟩}{⟨text⟩}

The command that produces the footnote. The default definition ignores the first argument:

\newcommand*{\glsxtrabbrvfootnote}[2]{\footnote{#2}}

\glsxtrfootnotelongformat{⟨entry-label⟩}{⟨fmt-cs⟩}

This command is used within the footnote to display the long form formatted with `⟨fmt-cs⟩` for the footnote styles on first use of `\gls`, `\GlS` and `\GLS`. The default definition is simply:
For example, if the footnote should start with an uppercase letter then simply redefine this to use \Glsxtrlongformat instead:

\renewcommand*{\glsxtrfootnotelongformat}[2]{\Glsxtrlongformat{#1}{\{#2}\}}

\glsxtrfootnotelongplformat{⟨entry-label⟩}{⟨fmt-cs⟩}

This command is used within the footnote to display the plural long form formatted with ⟨fmt-cs⟩ for the footnote styles on first use of \glspl, \Glspl and \GLSpl. The default definition is simply:

\newcommand*{\glsxtrfootnotelongplformat}[2]{\glsxtrlongplformat{#1}{\{#2}\}}

\glsxtrpostfootnotelongformat{⟨entry-label⟩}{⟨fmt-cs⟩}

This command is used for the “postfootnote” styles. This is simply defined to do \glsxtrfootnotelongformat. Note that there’s no plural equivalent as the “postfootnote” styles don’t check if the plural command (\glspl etc) was used.

4.5.1.3.5. No-Long Styles

These commands are used by the “nolong” styles.

\glsxtrshortnolongname

This command should expand to the value of the name key for styles like short-nolong. The default definition is:
4. Abbreviations

\glsxpabbrvfont{\the\glsshorttok}{\glscategorylabel}

\glsxtrshortdescname

This command should expand to the value of the \texttt{name} key for styles like \texttt{short-nolong-desc}. The default definition is:

\glsxpabbrvfont{\the\glsshorttok}{\glscategorylabel}\%
\protect\glsxtrfullsep{\the\glslabeltok}\%
\protect\glsxtrparen
{\glsxplongfont{\the\glslongtok}{\glscategorylabel}}\%

4.5.1.3.6. No-Short Styles

These commands are used by the “noshort” styles.

\glsxtrlongnoshortdescname

This command should expand to the value of the \texttt{name} key for styles like \texttt{long-noshort-desc}. The default definition is:

\glsxplongfont{\the\glslongtok}{\glscategorylabel}

\glsxtrlongnoshortname

This command should expand to the value of the \texttt{name} key for styles like \texttt{long-noshort}. The default definition is:

\glsxpabbrvfont{\the\glsshorttok}{\glscategorylabel}
4. Abbreviations

4.5.1.3.7. Hyphen Styles

These are commands used by the “hyphen” styles. They are designed to work with the markwords and markshortwords attributes.

\glsabbrvhyphenfont{⟨text⟩}

The formatting command used for the short form in the hyphen styles. The default is to simply use \glsabbrvdefaultfont.

\glsfirstabbrvhyphenfont{⟨text⟩}

The formatting command used for the short form in the hyphen styles on first use. The default is to simply use \glsabbrvhyphenfont.

\glslonghyphenfont{⟨text⟩}

The formatting command used for the long form in the hyphen styles. The default is to simply use \glslongdefaultfont.

\glsfirstlonghyphenfont{⟨text⟩}

The formatting command used for the long form in the hyphen styles on first use. The default is to simply use \glslonghyphenfont.

\glsxtrhyphensuffix  \textit{initial:} \glsxtrabbrvpluralsuffix

Short plural suffix used by the “hyphen” styles. This defaults to \glsxtrabbrvplural-suffix.

\glsxtrlonghyphenshortsort

Expands to the sort value for the styles like long-hyphen-short-hyphen. This defaults to the original short value (\glsxtrorgshort). This command is irrelevant with the “unsrt” family of commands.

\glsxtrshorthyphenlongsort
Expands to the sort value for the styles like \textit{short-hyphen-long-hyphen}. This defaults to the original short value (\glsxtrorgshort). This command is irrelevant with the “unsrt” family of commands.

\glsxtrlonghyphennoshortsort

Expands to the sort value for the styles like \textit{long-hyphen-noshort-noreg}. This defaults to the original short value (\glsxtrorgshort). This command is irrelevant with the “unsrt” family of commands.

\glsxtrlonghyphennoshortdescsort

Expands to the sort value for the styles like \textit{long-hyphen-noshort-desc-noreg}. This defaults to the original long value (\glsxtrorglong). This command is irrelevant with the “unsrt” family of commands.

\glsxtrlonghyphenshort{⟨entry-label⟩}{⟨long⟩}{⟨short⟩}{⟨insert⟩}

Formats the long and short form for the full or first use \textit{long-hyphen-short-hyphen} style. This uses \glsxtrifhyphenstart to test if the ⟨insert⟩ starts with a hyphen. If it does, \glsxtrwordsep is locally set to \glsxtrwordsephyphen to replace the inter-word spaces with hyphens. The short form is placed in parentheses with \glsxtrparen, preceded by the \glsxtrfullsep separator. The ⟨insert⟩ is placed after both the long and the short form.

\GLSxtrlonghyphenshort{⟨entry-label⟩}{⟨long⟩}{⟨short⟩}{⟨insert⟩}

As above, but the ⟨insert⟩ is converted to all caps. The ⟨short⟩ and ⟨long⟩ arguments should be supplied as all caps. Note that it’s not possible to simply do \glsxtrlonghyphenshort with \MakeUppercase{⟨insert⟩} as the argument as this will interfere with the check to determine if ⟨insert⟩ starts with a hyphen.

\glsxtrlonghyphennoshort{⟨entry-label⟩}{⟨long⟩}{⟨insert⟩}

Formats the long form for the full or first use \textit{long-hyphen-noshort-desc-noreg} style. This uses \glsxtrifhyphenstart to test if the ⟨insert⟩ starts with a hyphen. If it does, \glsxtrwordsep is locally set to \glsxtrwordsephyphen to replace the inter-word spaces with
4. Abbreviations

hyphens. The \langle insert\rangle is placed after the long form.

\GLSxtrlonghyphennoshort{\langle entry-label\rangle}{\langle long\rangle}{\langle insert\rangle}

As above but converts \langle insert\rangle to all caps. The \langle long\rangle argument should already be in all caps. Note that it’s not possible to simply do \glsxtrlonghyphennoshort with \MakeUppercase {\langle insert\rangle} as the argument as this will interfere with the check to determine if \langle insert\rangle starts with a hyphen.

\glsxtrlonghyphen{\langle entry-label\rangle}{\langle long\rangle}{\langle insert\rangle}

Formats the long form for the full or first use long–hyphen–postshort–hyphen style. This is similar to the above, but the \langle insert\rangle argument is only used to check if it starts with a hyphen. The actual \langle insert\rangle is placed in the post-link hook.

\xpglsxtrposthyphenshort

This command is used in the post-link hook for the long–hyphen–postshort–hyphen style on first use. It expands the placeholder commands (\glslabel and \glsinsert) and uses \GLSxtrposthyphenshort for all caps or \glsxtrposthyphenshort otherwise. Note that this doesn’t show the plural by default. If you require the plural form, you need to redefine this to add a check with \glsifplural:

\newcommand*{\xpglsxtrposthyphenshort}{% \
  \glsifplural
  {\% \glsxtrifallcaps
    {\% \expandafter\GLSxtrposthyphenshortpl\expandafter\glslabel
      \expandafter{{\glsinsert}}%
    {\% \expandafter\glsxtrposthyphenshortpl\expandafter\glslabel
      \expandafter{{\glsinsert}}%
    {\% \glsxtrifallcaps
      {\%
    {\%
  \% \glsxtrifallcaps
  {\%}
4. Abbreviations

\begin{verbatim}
\expandafter\GLSxtrposthyphenshort\expandafter\glslabel
\expandafter{\glsinsert}\
\expandafter\glsxtrposthyphenshort\expandafter\glslabel
\expandafter{\glsinsert}
\end{verbatim}

If \texttt{\{insert\}} starts with a hyphen, \texttt{\glsxtrwordsep} is locally set to \texttt{\glsxtrwordsephyphen} to replace the inter-word spaces with hyphens. The \texttt{\{insert\}} encapsulated with \texttt{\glsfirstlonghyphenfont} is then done (to complete the long form, which has already been displayed with \texttt{\glslonghyphen} in the link text). Then the short form followed by the \texttt{\{insert\}} is placed in parentheses (with \texttt{\glsxtrparen} preceded by \texttt{\glsxtrfullsep}).

\begin{verbatim}
\glsxtrposthyphenshort\{\texttt{\{entry-label\}}\}\{\texttt{\{insert\}}\}
\end{verbatim}

As above but all caps.

\begin{verbatim}
\glsxtrposthyphenshortpl\{\texttt{\{entry-label\}}\}\{\texttt{\{insert\}}\}
\end{verbatim}

As \texttt{\glsxtrposthyphenshort} but plural.

\begin{verbatim}
\GLSxtrposthyphenshortpl\{\texttt{\{entry-label\}}\}\{\texttt{\{insert\}}\}
\end{verbatim}

As above but all caps.

\texttt{xpglsxtrposthyphensubsequent}

This command is used in the post-link hook for the \texttt{long-hyphen-postshort-hyphen} style on subsequent use. It expands the placeholder commands (\texttt{\glslabel} and \texttt{\glsinsert}) and uses \texttt{\GLSxtrposthyphensubsequent} for all caps or \texttt{\glsxtrposthyphensubsequent}.
4. Abbreviations

otherwise.

\glsxtrposthyphensubsequent\{entry-label\}\{insert\}

This command is used in the post-link hook for the long-hyphen-postshort-hyphen style on subsequent use. Only the \{insert\} is done.

\GLSxtrposthyphensubsequent\{entry-label\}\{insert\}

As above but all caps.

\glsxtrshorthyphenlong\{entry-label\}\{short\}\{long\}\{insert\}

Formats the short and long form for the full or first use short-hyphen-long-hyphen style. Similar to \glsxtrlonghyphenshort but the short and long forms are swapped.

\GLSxtrshorthyphenlong\{entry-label\}\{short\}\{long\}\{insert\}

As above, but the \{insert\} is converted to all caps. The \{short\} and \{long\} arguments should be supplied as all caps. Note that it’s not possible to simply do \glsxtrshorthyphenlong with \MakeUppercase\{insert\} as the argument as this will interfere with the check to determine if \{insert\} starts with a hyphen.

\glsxtrshorthyphen\{short\}\{entry-label\}\{insert\}

Formats the short form for the full or first use short-hyphen-postlong-hyphen style. The \{insert\} argument is only used to check if it starts with a hyphen. The actual \{insert\} is placed in the post-link hook.

\xpglsxtrposthyphenlong

This command is used in the post-link hook for the short-hyphen-postlong-hyphen style on first use. It expands the placeholder commands (\glslabel and \glsinsert) and uses \GLSxtrposthyphenlong for all caps or \glsxtrposthyphenlong otherwise. Note that this doesn’t show the plural by default. If you require the plural form, you need to redefine this to add a check with \glsifplural:
\newcommand*{\xpglsxtrpostphyphenlong}{\%
\glsifplural
\%
\glsxtrifallcaps
\%
\expandafter\GLSxtrposthyphenlongpl\expandafter\glslabel
\expandafter\{\glsinsert\%
\%
\expandafter\glsxtrposthyphenlongpl\expandafter\glslabel
\expandafter\{\glsinsert\%
\%
\expandafter\glsxtrposthyphenlongpl\expandafter\glslabel
\expandafter\{\glsinsert\%
\%
\expandafter\glsxtrposthyphenlongpl\expandafter\glslabel
\expandafter\{\glsinsert\%
\%
\}
\%
}
\glsxtrposthyphenlong\{\langle entry-label\rangle\}{\langle insert\rangle}

This command is used in the post-link hook for the short-hyphen-postlong-hyphen style on first use. Similar to \glsxtrpostphyphenshort but shows the long form instead of the short form.

\GLSxtrpostphyphenlong\{\langle entry-label\rangle\}{\langle insert\rangle}

As above but all caps.

\glsxtrpostphyphenlongpl\{\langle entry-label\rangle\}{\langle insert\rangle}
4. Abbreviations

As \glsxtrposthyphenlong but plural.

\GLSxtrposthyphenlongpl\{entry-label\}\{insert\}

As above but all caps.

4.5.1.3.8. Only Styles

These are commands used by the “only” styles, such as \textit{long-only-short-only}.

\glsabbrvonlyfont\{text\}

The formatting command used for the short form in the only styles. The default is to simply use \glsabbrvdefaultfont.

\glsfirstabbrvonlyfont\{text\}

The formatting command used for the short form in the only styles on first use. The default is to simply use \glsabbrvonlyfont.

\glslongonlyfont\{text\}

The formatting command used for the long form in the only styles. The default is to simply use \glslongdefaultfont.

\glsfirstlongonlyfont\{text\}

The formatting command used for the long form in the only styles on first use. The default is to simply use \glslongonlyfont.

\glsxtronlysuffix initial: \glsxtrabbrvpluralsuffix

Short plural suffix used by the “only” styles. This defaults to \glsxtrabbrvpluralsuffix.

\glsabbrvsconlyfont\{text\}
4. Abbreviations

The formatting command used for the short form in the “sc-only” styles. The default is to simply use \glsabbrvscfont.

\glsfirstabbrvsconlyfont{⟨text⟩}

The formatting command used for the short form in the “sc-only” styles on first use. The default is to simply use \glsabbrvsconlyfont.

\glsxtrsconlyrevert{⟨text⟩}

Counteracts the effect of \glsabbrvsconlyfont. The default is \glsxtrscrevert. If you redefine \glsabbrvsconlyfont, you will need to redefine \glsxtrsconlyrevert as applicable.

\glsxtrsonlysuffix

Initial: \glsxtrscsuffix

Short plural suffix used by the “sc-only” styles. This defaults to \glsxtrscsuffix.

\glsxtronlyname

Expands to the value for the name key for the “only” styles. The default definition is:

\protect\glsabbrvonlyfont{\the\glsshorttok}

\glsxtronlydescname

Expands to the value for the name key for the “only” styles where the description should be described, such as long-only-short-only-desc. The default definition is:

\protect\glslongfont{\the\glslongtok}

\glsxtronlydescsort

155
4. Abbreviations

Expands to the value for the sort key for the “only” styles where the description should be described, such as long-only-short-only-desc. The default definition is \the\glslongtok.

This command is irrelevant with the “unsrt” family of commands.

\glsxtrsconlyname

Expands to the value for the name key for the “sc-only” styles. The default definition is:

\protect\glsabbrvsconlyfont{\the\glsshorttok}

\glsxtrsconlydescname

Expands to the value for the name key for the “sc-only” styles where the description should be described. The default definition is to simply use \glsxtrsconlydescname.

\glsxtrsconlydescsort

Expands to the value for the sort key for the “sc-only” styles where the description should be described, such as long-only-short-only-desc. The default definition is to simply use \glsxtrsconlydescsort.

This command is irrelevant with the “unsrt” family of commands.

4.5.1.3.9. Fonts

These are commands used by styles that use a particular font shape or size, identified by one of the following two-letter tags: “sc” (\textsc), “sm” (\textsmaller) or “em” (\emph).

For the “sc-user” styles, see §4.5.1.3.3. For the “sc-only” styles, see §4.5.1.3.8.

\glsabbrvscfont\{text\}
4. Abbreviations

Formatting for the “sc” short form. This uses \textsc to apply a small caps style, so your document font needs to support it.

\textsc uses small capital glyphs for lowercase characters. Uppercase characters show as normal capitals. This means that you need to use lowercase characters in the abbreviation.

\glsfirstabbrvscfont\{⟨text⟩\}

Formatting for the “sc” short form shown on first use. This defaults to \glsabbrvscfont.

\glsxtrscrevert\{⟨text⟩\}

Counteracts the effect of \glsabbrvscfont. This defaults to \glstextup. If you redefine \glsabbrvscfont, you will need to redefine \glsxtrscrevert as applicable.

\glsxtrscsuffix

Short plural suffix used by the “sc” styles. This needs to counteract the smallcaps, so it’s defined as:

\protect\glstextup\glsxtrabbrvpluralsuffix

\glsabbrvsmfont\{⟨text⟩\}

Formatting for the “sm” short form. This uses \textsmaller, which is defined by the relsize package. You will need to load that package if you want to use any of the “sm” styles.

\textsmaller reduces the font size, so if you want to use it to simulate small caps, you need to use uppercase characters in the abbreviation.

\glsfirstabbrvsmfont\{⟨text⟩\}
4. Abbreviations

Formatting for the “sm” short form shown on first use. This defaults to \glsabbrvsmfont.

\glsxtrsmrevert{(text)}

Counteracts the effect of \glsabbrvsmfont. This defaults to \textlarger. If you redefine \glsabbrvsmfont, you will need to redefine \glsxtrsmrevert as applicable.

\glsxtrmsuffix initial: \glsxtrabbrvpluralsuffix

Short plural suffix used by the “sm” styles. This defaults to \glsxtrabbrvpluralsuffix.

\glsabbrvemfont{(text)}

Formatting for the “em” short form. This uses \emph.

\glsfirstabbrvemfont{(text)}

Formatting for the “em” short form shown on first use. This defaults to \glsabbrvemfont.

\glsxtremrevert{(text)}

Counteracts the effect of \glsabbrvemfont. This defaults to \textup. If you redefine \glsabbrvemfont, you will need to redefine \glsxtremrevert as applicable.

\glsxtremsuffix initial: \glsxtrabbrvpluralsuffix

Short plural suffix used by the “em” styles. This defaults to \glsxtrabbrvpluralsuffix.

\glslongemfont{(text)}

Formatting for the “em” long form. This uses \emph.

\glsfirstlongemfont{(text)}

Formatting for the “em” short form shown on first use. This defaults to \glslongemfont.
4. Abbreviations

4.5.2. Advanced Style Commands

These commands should typically not be needed in a document, but are provided for advanced users. See §4.5.1.3 for commands to adjust the predefined abbreviation styles.

\glssetabbrvfmt{<category>}

Sets the current formatting commands (§4.5.3.2) associated with the abbreviation style associated with the given category. That is, the command redefinitions provided in the third argument (\emph{display definitions}) of \newabbreviationstyle are applied.

If no abbreviation style has been set for the given category, the style associated with the \texttt{abbreviation} category is used.

This command is used:

- At the start of \glsentryfmt if the current entry has the \texttt{short} field set. This ensures that the \gls-like commands use the appropriate formatting.

- At the start of \glsxtrassignfieldfont if the current entry has the \texttt{short} field set. This ensures that the \gls-like commands use the appropriate formatting (where possible).

- At the start of \glsxtrshort, \glsxtrlong, \glsxtrfull and their plural and case-changing variants.

- At the start of \glossentryname, \glossentrynameother, \glossentrydesc, \Glossentryother, \glossentrydesc, \glossentrysymbol and \Glossentrysymbol.

\glsuseabbrvfont{<style-name>}{<text>}

A robust command that applies the abbreviation font for the given category to the supplied text.

\glsuselongfont{<style-name>}{<text>}

A robust command that applies the long font for the given category to the supplied text.

\GlsXtrUseAbbrStyleSetup{<style-name>}

159
4. Abbreviations

This implements the given abbreviation style’s setup code. Note that this expects the placeholder macros and token registers to be set. This may be used in the \setup of \newabbreviationstyle to inherit the setup code of a related style.

\GlsXtrUseAbbrStyleFmts{(style-name)}

This implements the given abbreviation style’s display definitions code. This may be used in the \display definitions of \newabbreviationstyle to inherit the formatting of a related style.

\xpglsxtrpostabbrvfootnote

This is used by styles like postfootnote to ensure that the label and inner and outer formatting are expanded before being passed to \glsxtrpostabbrvfootnote, otherwise they may lose their definitions before the footnote text is typeset.

\glsxtrpostabbrvfootnote{(entry-label)}{(fmt-code)}

This is used by the footnote styles that defer the footnote to the post-link hook. The default definition is:

\newrobustcmd*{\glsxtrpostabbrvfootnote}[2]{% \glsxtrabbrvfootnote{#1}{#2\glsxtrpostfootnotelongformat{#1}{\glsfirstlongfootnotefont}}}%

The second argument will be the expansion of \glstrassignlinktextfmt, to allow the inner formatting to be picked up, if required.

\glsxtrifhyphenstart{(text)}{(true)}{(false)}

This command is used by the hyphen styles to determine if the insert material starts with a hyphen. Does \true if \text starts with a hyphen otherwise does \false.

\GlsXtrWarnDeprecatedAbbrStyle{(old-name)}{(new-name)}

This command is used to generate a warning (with \GlossariesExtraWarning) if a deprecated abbreviation style is used.
4. Abbreviations

4.5.3. Defining New Abbreviation Styles

If none of the predefined styles suit your requirements, you can define your own custom style using:

\newabbreviationstyle{⟨style-name⟩}{⟨setup⟩}{⟨display definitions⟩}

The first argument is the style name. This is used internally to form control sequences, so the name shouldn’t contain any special characters.

The second argument sets up the information that’s required when an abbreviation is defined (which is why the style must be set before the abbreviations with that style are defined). The relevant commands for this argument are listed in §4.5.3.1.

The third argument defines the commands that determine how the display style (\gls) and the inline style (\glsxtrfull) are formatted. The relevant commands for this argument are listed in §4.5.3.2.

\renewabbreviationstyle{⟨style-name⟩}{⟨setup⟩}{⟨display definitions⟩}

Redefines an existing abbreviation style.

\letabbreviationstyle{⟨new style⟩}{⟨existing style⟩}

Defines a synonym of an existing abbreviation style.

4.5.3.1. Style Initialisation Hooks

The style initialisation hooks should be placed in the second argument (⟨setup⟩) of \newabbreviationstyle. They ensure that all the fields are correctly initialised when the entry is defined with the underlying \newglossaryentry command. They may also be used to set category attributes.

The following is prepended to ⟨setup⟩ to initialise the final hook:

\renewcommand*{\GlsXtrPostNewAbbreviation}{}

When an entry is defined with \newabbreviation, the following steps are performed:

1. Token registers are initialised to the information provided in the arguments of \newabbreviation: \glskeylisttok, \glslabeltok, \glsshorttok and \glslongtok.

2. The commands \glsxtrorgkeylist, \glsxtrorgshort and \glsxtrorglong are defined to the options, short and long values supplied to \newabbreviation. (The \glskeylisttok \glsshorttok and \glslongtok token registers may be changed
4. Abbreviations

before the entry is actually defined. These commands may be used to obtain the original values.)

3. \texttt{\textbackslash ExtraCustomAbbreviationFields} is initialised to do nothing.

4. Accessibility settings are initialised, if required. These redefine \texttt{\textbackslash ExtraCustomAbbreviationFields} to set the accessibility fields.

5. The command \texttt{\textbackslash glscategorylabel} is defined to abbreviation.

6. The options list is parsed for the following keys: category and, if accessibility is enabled, access, textaccess, pluralaccess, firstaccess, firstpluralaccess, shortaccess, shortpluralaccess, longaccess, and longpluralaccess.

7. The abbreviation style is applied for the category given by \texttt{\textbackslash glscategorylabel} (which may have been changed when the options were parsed in the previous step) or the fallback if no abbreviation style is associated with that category. This performs both the \texttt{(setup)} and \texttt{(display definitions)} provided when the style was defined with \texttt{\textbackslash newabbreviationstyle}.

8. The long plural form is initialised to its default value (\texttt{\textbackslash long\textbackslash glspluralsuffix}).

9. The \texttt{markwords} attribute, if set, is implemented for the singular long form. It will also mark the entry as having a description with formatting (using \texttt{\textbackslash glsexclapplyinnerfmtfield}).

10. The \texttt{markshortwords} attribute is implemented, if set, otherwise the \texttt{insertdots} attribute is implemented, if set, for the singular short form.

11. The \texttt{aposplural} attribute is implemented, if set, otherwise the \texttt{noshortplural} attribute is implemented, if set. This step will set the default short plural.

12. \texttt{\textbackslash glsshorttok} is updated to reflect any changes.

13. The \texttt{\textbackslash glsxtrnewabbrevpresetkeyhook} hook is performed.

14. The options list is parsed for the \texttt{shortplural} and \texttt{longplural} keys. The \texttt{\textbackslash glskeylisttok} token is updated to only include the remaining keys that haven’t yet been processed.

15. The \texttt{markwords} attribute, if set, is implemented for the plural long form.

16. The \texttt{markshortwords} attribute, if set, otherwise the \texttt{insertdots} attribute, if set, is implemented for the plural short form.

17. The \texttt{\textbackslash glsshortpltok} and \texttt{\textbackslash glslongpltok} registers are set.

18. \texttt{\textbackslash newabbreviationhook} performed.

19. The entry is defined using \texttt{\textbackslash newglossaryentry} with the key value list:
4. Abbreviations

20. Add the name, first, firstplural, text and plural keys to the list of inner formatting exclusions, as they include formatting commands.

21. Final hook \GlsXtrPostNewAbbreviation performed.

Note that when these hooks (except the last) are used, the entry hasn’t yet been defined. However, some information will have already been picked up from the arguments of \newabbreviation. These can be accessed in the hooks using the following (but make sure they are fully expanded):

- \glscategorylabel
  Expands to the entry’s category label.

- \glskeylisttok
  A token register that contains the options that were passed to \newabbreviation with preprocessed options removed. Use \the\glskeylisttok to expand it.
  The original option list, as supplied to \newabbreviation, can be obtained with:

- \glsxtrorgkeylist
  (Not a token register.)

- \glslabeltok
4. Abbreviations

A token register that contains the entry’s label. Use \the\glslabeltok to expand it.

\glslabeltok

A token register that contains the short form (which may have been modified after being passed to \newabbreviation). Use \the\glsshorttok to expand it.

The original short form, as supplied to \newabbreviation, can be obtained with:

\glsxtrorgshort

(Not a token register.)

\glsshortpltok

A token register that contains the short plural form (which may have been obtained from the short form or modified after being passed to \newabbreviation). Use \the\glsshortpltok to expand it.

\glslongtok

A token register that contains the long form (which may have been modified after being passed to \newabbreviation). Use \the\glslongtok to expand it.

The original long form, as supplied to \newabbreviation, can be obtained with:

\glsxtrorglong

(Not a token register.)

\glslongpltok

A token register that contains the long plural form (which may have been obtained from the long form or modified after being passed to \newabbreviation). Use \the\glslongpltok to expand it.

\ExtraCustomAbbreviationFields

Expands to additional field definitions for the entry. This is used to add the accessibility fields (such as shortaccess), if enabled. The abbreviation style may append (\appto) or prepend
4. Abbreviations

\(\text{\textbackslash preto}\) additional information, if required, to this hook.

If you alter this hook, make sure that you include the trailing comma after each \(\langle key\rangle = \{value\}\), including the last one.

\CustomAbbreviationFields

Expands to the default field definitions for the entry. Take care to protect any commands that shouldn’t be expanded. The comma may be omitted from the final \(\langle key\rangle = \{value\}\).

\GlsXtrPostNewAbbreviation

A hook that’s used after the entry has been defined (at the end of \texttt{\newabbreviation}). This can be used to set category attributes, define the post-link hook, or mark the entry as having a complex style (with \texttt{\glsxtrsetcomplexstyle}).

For example, the \texttt{long-short} abbreviation style includes the following in \texttt{\setup}:

\renewcommand*{\GlsXtrPostNewAbbreviation}{% \glsxtrsetcomplexstyle{\the\glslabeltok}{3} \glshasattribute{\the\glslabeltok}{regular}% {% \glssetattribute{\the\glslabeltok}{regular}{false}% }% {%} %}

Note that in the above, the commands within the definition of \texttt{\GlsXtrPostNewAbbreviation} are all expanded when that hook is used. However, if this hook defines other commands or hooks that will be used later, then make sure that the definitions of those commands use the inner hook’s own placeholder commands.

Remember that the post-link hook uses \texttt{\glslabel} to reference the current label. Don’t use \texttt{\glslabeltok} as that will contain the label of the last abbreviation to be defined.

For example, the \texttt{long-hyphen-postshort-hyphen} style has:

165
In the above, \glslabeltok and \glscategorylabel are used in the parts that will be expanded at the end of \newabbreviation, but \glslabel and \glsinsert are used in the definition of the post-link hook, which won’t be expanded until the entry is referenced in the document with a command such as \gls. (The use of \expandafter is included to assist \innertextformat.)

\glsxtrsetcomplexstyle{⟨entry-label⟩}{⟨n⟩}

This command should go in the definition of \GlsXtrPostNewAbbreviation to indicate that the entry given by ⟨entry-label⟩ has an abbreviation style that is complex. The second argument ⟨n⟩ should be numeric and indicates why it doesn’t work with \glsfirst, \Glsfirst, \GLSfirst, \glsfirstplural, \Glsfirstplural or \GLSfirstplural: 1 (all caps doesn’t work), 2 (all caps and insert doesn’t work), 3 (insert doesn’t work).

\glsfirstinnerfmtabbrvfont{⟨text⟩}

This is a robust command that applies both \glsfirstabbrvfont and the inner formatting
4. Abbreviations

command \glsxtrgenentrytextfmt. This is used by the following command.

\glsfirstxpabbvrfont{(text)}{(category)}

If the markshortwords attribute is true, this does \protect\glsfirstabbvrfont{(text)} otherwise it does \glsfirstinnerfmtabbvrfont{(text)}.

This command is designed to be used within \CustomAbbreviationFields to set the first and firstplural keys, so it needs to partially expand within \newabbreviation. For example, the postfootnote includes the following lines in the definition of \CustomAbbreviationFields:

\begin{verbatim}
first={\glsfirstxpabbvrfont{\the\glsshorttok}{\glscategorylabel}},%
firstplural={\glsfirstxpabbvrfont{\the\glsshortpltok}{\glscategorylabel}},
\end{verbatim}

This will be expanded before being passed to \newglossaryentry. If the markshortwords attribute is true, this will end up as:

\begin{verbatim}
first={\protect\glsfirstabbvrfont{(short)}},%
firstplural={\protect\glsfirstabbvrfont{(shortpl)}},%
\end{verbatim}

otherwise it will end up as:

\begin{verbatim}
first={\glsfirstinnerfmtabbvrfont{(short)}},%
firstplural={\glsfirstinnerfmtabbvrfont{(shortpl)}},%
\end{verbatim}

where (short) and (shortpl) are, respectively, the values in the \glsshorttok and \glsshortpltok registers.

The placeholder registers and macros (such as \glsshorttok and \glscategorylabel) must be expanded before being passed to \newglossaryentry as their values are unreliable outside of \newabbreviation.

\glsinnerfmtabbvrfont{(text)}

This is a robust command that applies both \glsabbvrfont and the inner formatting com-
4. Abbreviations

mand \texttt{\textbackslash glsxtrgenentrytextfmt}. This is used by the following command.

\begin{verbatim}
\texttt{\textbackslash glsxpabbrvfont\{\langle text\rangle\}\{\langle category\rangle\}}
\end{verbatim}

If the \texttt{markshortwords} attribute is true, this does \texttt{\textbackslash protect\textbackslash glsabbrvfont\{\langle text\rangle\}} otherwise it does \texttt{\textbackslash glsinnerfmtabbrvfont\{\langle text\rangle\}}. This command is designed for the \texttt{name}, \texttt{text} and \texttt{plural} keys within \texttt{\textbackslash CustomAbbreviationFields}.

\begin{verbatim}
\texttt{\textbackslash glsfirstinnerfmtlongfont\{\langle text\rangle\}}
\end{verbatim}

This is a robust command that applies both \texttt{\textbackslash glsfirstlongfont} and the inner formatting command \texttt{\textbackslash glsxtrgenentrytextfmt}. This is used by the following command.

\begin{verbatim}
\texttt{\textbackslash glsxplongfont\{\langle category\rangle\}\{\langle text\rangle\}}
\end{verbatim}

If the \texttt{markwords} attribute is true, this does \texttt{\textbackslash protect\textbackslash glsfirstlongfont\{\langle text\rangle\}} otherwise it does \texttt{\textbackslash glsfirstinnerfmtlongfont\{\langle text\rangle\}}. This command is designed for the \texttt{first} and \texttt{firstplural} keys within \texttt{\textbackslash CustomAbbreviationFields}.

\begin{verbatim}
\texttt{\textbackslash glsinnerfmtlongfont\{\langle text\rangle\}}
\end{verbatim}

This is a robust command that applies both \texttt{\textbackslash glslongfont} and the inner formatting command \texttt{\textbackslash glsxtrgenentrytextfmt}. This is used by the following command.

\begin{verbatim}
\texttt{\textbackslash glsxplongfont\{\langle category\rangle\}\{\langle text\rangle\}}
\end{verbatim}

If the \texttt{markwords} attribute is true, this does \texttt{\textbackslash protect\textbackslash glslongfont\{\langle text\rangle\}} otherwise it does \texttt{\textbackslash glsinnerfmtlongfont\{\langle text\rangle\}}. This command is designed for the \texttt{name}, \texttt{text} and \texttt{plural} keys within \texttt{\textbackslash CustomAbbreviationFields} (if they should include the long form in their value, such as the \texttt{long-noshort-desc} style).

\begin{verbatim}
\texttt{\textbackslash glsxtrAccSuppAbbrSetNoLongAttrs\{\langle category\rangle\}}
\end{verbatim}

If accessibility support has been enabled with \texttt{accsupp}, this command will initialise support for the \texttt{name}, \texttt{first}, \texttt{firstplural}, \texttt{text} and \texttt{plural} fields for the given category (using \texttt{\textbackslash glsxtrprovideaccsuppcmd}). The \texttt{nameshortaccess}, \texttt{firstshortaccess} and \texttt{textshortaccess} attributes are set to \texttt{true}. (Does nothing if accessibility support has not been enabled.)
This command is provided for abbreviation styles where the name, first and text are just the formatted abbreviation. The first field may just be the long form or may be a combination of short and long.

\glsxtrAccSuppAbbrsetNameLongAttrs{(category)}

If accessibility support has been enabled with accsupp, this command will initialise support for the first, firstplural, text and plural fields for the given category (using \glsxtrprovideaccsuppcmd). The firstshortaccess and textshortaccess attributes are set to true. (Does nothing if accessibility support has not been enabled.)

This command is provided for abbreviation styles where the first and text are just the formatted abbreviation. The name field may just be the long form or may be a combination of short and long.

\glsxtrAccSuppAbbrsetFirstLongAttrs{(category)}

If accessibility support has been enabled with accsupp, this command will initialise support for the name, text and plural fields for the given category (using \glsxtrprovideaccsuppcmd). The nameshortaccess and textshortaccess attributes are set to true. (Does nothing if accessibility support has not been enabled.)

This command is provided for abbreviation styles where the text is just the formatted abbreviation. The name and first fields may just be the long form or may be a combination of short and long. The name may also be short but followed by the long form in the description.

\glsxtrAccSuppAbbrsetTextShortAttrs{(category)}

If accessibility support has been enabled with accsupp, this command will initialise support for the text and plural fields for the given category (using \glsxtrprovideaccsuppcmd). The textshortaccess attribute is set to true. (Does nothing if accessibility support has not been enabled.)

This command is provided for abbreviation styles where the name field for the given category (using \glsxtrprovideaccsuppcmd). The nameshortaccess attribute is set to true. (Does nothing if accessibility support has not been enabled.)
This command is provided for abbreviation styles where only the name is just the formatted abbreviation. The first and text fields may just be the long form or may be a combination of short and long.

### 4.5.3.2. Style Formatting Commands

The final \textit{⟨display definitions⟩} argument of \texttt{\newabbreviationstyle} should contain the redefinitions of the style commands listed here that are used to format abbreviations.

Whenever an abbreviation style is activated with commands like \texttt{\setabbreviationstyle}, \texttt{\newabbreviation} or \texttt{\glssetabbrvfmt}, \textit{⟨display definitions⟩} are implemented.

If you simply want to adjust the formatting of one of the predefined styles, you should redefine the associated commands listed in §4.5.1.3.

The following initialisation is always prepended to \textit{⟨display definitions⟩} so you can omit them if the default is appropriate for your style:

\begin{verbatim}
\renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}\
\renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}\
\renewcommand*{\GLSxtrinlinefullformat}{\GLSxtrfullformat}\
\renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}\
\renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}\
\renewcommand*{\GLSxtrinlinefullplformat}{\GLSxtrfullplformat}\
\let\glsxtrsubsequentfmt\glsxtrdefaultsubsequentfmt\
\let\glsxtrsubsequentplfmt\glsxtrdefaultsubsequentplfmt\
\let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt\
\let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt\
\let\GLSxtrsubsequentfmt\GLSxtrdefaultsubsequentfmt\
\let\GLSxtrsubsequentplfmt\GLSxtrdefaultsubsequentplfmt
\end{verbatim}

In the event that any styles omit defining the newer \texttt{\GLSxtrfullformat} or \texttt{\GLSxtrfullplformat}, these are also initialised to defaults but ideally they should have their definitions provided.

The minimal set of commands that should have their definitions provided are the abbreviation plural suffix (\texttt{\abbrvpluralsuffix}) the display full forms: \texttt{\glsxtrfullformat}, \texttt{\glsxtrfullplformat} and their case-changing variants.

The inline full form commands only need to be provided if they behave differently from the display full form. The subsequent use commands only need to be provided if the default (only show the short form) isn’t suitable.
4. Abbreviations

The content of \emph{display definitions} is placed within the definition of an internal control sequence, so remember to use \texttt{\#\#} instead of \texttt{\#} to reference command parameters.

4.5.3.2.1. Suffix and Fonts

These are the generic suffix and font commands that vary according to the abbreviation style. The style should provide the appropriate definitions. The suffix should always be provided. The font commands are only required if the style applies any font formatting to either the long or short form.

\texttt{\abbrvpluralsuffix} \\
\textit{initial:} \texttt{\glsxtrabbrvpluralsuffix}

The plural suffix for the short form. For example, the \texttt{long-short} style defines this to just use \texttt{\glsxtrabbrvpluralsuffix}, but the smallcaps styles, such as \texttt{long-short-sc} define this to \texttt{\glsxtrscsuffix} in order to counteract the small caps font.

\texttt{\glsfirstabbrvfont\{\text\}}

The font formatting command for the short form on first use. For example, the \texttt{long-short-sc} style has:

\texttt{\renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont\#1}}

\texttt{\glsabbrvfont\{\text\}}

The font formatting command for the short form. For example, the \texttt{long-short-sc} style has:

\texttt{\renewcommand*\glsabbrvfont[1]{\glsabbrvscfont\#1}}

\texttt{\glsxtrrevert\{\text\}}

This command is designed to counteract the effect of \texttt{\glsabbrvfont} if, for some reason, it shouldn’t be applied to part of the abbreviation. For example, you may prefer not to have
digits reduced with the smaller (“sm”) styles.

\glsfirstlongfont{(text)}

The font formatting command for the long form on first use. For example, the \texttt{long-short-sc} style has:

\renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}

\glslongfont{(text)}

The font formatting command for the long form. For example, the \texttt{long-short-sc} style has:

\renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}

4.5.3.2.2. First Use Display Format
These commands always need to be provided.

\glsxtrfullformat{(entry-label)}{(insert)}

The singular display full form used on the first use of \gls.

\glsxtrfullplformat{(entry-label)}{(insert)}

The plural display full form used on the first use of \glspl.

\Glsxtrfullformat{(entry-label)}{(insert)}

The sentence case singular display full form used on the first use of \Gls.
4. Abbreviations

The sentence case plural display full form used on the first use of \Glspl.

\GLSxtrfullformat{⟨entry-label⟩}{⟨insert⟩}

The all caps singular display full form used on the first use of \GLS.

\GLSxtrfullplformat{⟨entry-label⟩}{⟨insert⟩}

The all caps plural display full form used on the first use of \GLSp1.

4.5.3.2.3. Subsequent Use Display Format

These commands only need to be provided if the \gls-like commands don’t simply show the short form.

\glsxtrsubsequentfmt{⟨entry-label⟩}{⟨insert⟩}

The singular form for subsequent use of \gls.

\glsxtrsubsequentplfmt{⟨entry-label⟩}{⟨insert⟩}

The plural form for subsequent use of \Glspl.

\Glsxtrsubsequentfmt{⟨entry-label⟩}{⟨insert⟩}

The sentence case singular form for subsequent use of \Gls.

\Glsxtrsubsequentplfmt{⟨entry-label⟩}{⟨insert⟩}

The sentence case plural form for subsequent use of \Glspl.

\GLSxtrsubsequentfmt{⟨entry-label⟩}{⟨insert⟩}

The all caps singular form for subsequent use of \GLS.

\GLSxtrsubsequentplfmt{⟨entry-label⟩}{⟨insert⟩}
The all caps plural form for subsequent use of \GLSpl.

The defaults all show the short form and insert encapsulated with the inner formatting \glsxtrgenentrytextfmt and \glsabbrvfont. The purpose of the inner formatting is to get it as close as possible to the actual text so \glsabbrvfont is placed outside of \glsxtrgenentrytextfmt.

The \texttt{markshortwords} attribute complicates matters as it inserts \glsxtrword and \glsxtrwordsep into the actual field value. In that case, the inner formatting is within \glsxtrword and \glsxtrwordsep, so only the insert material needs to be formatted.

If a custom style doesn’t need to support \texttt{innertextformat} or \texttt{ifglsxtrinsertinside}, it can reduce the complexity by omitting the inner formatting and conditionals, but this lack of support should be documented if the style is made generally available.

```
\glsxtrdefaultsubsequentfmt\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}
```

The default singular form for subsequent use of \texttt{\gls}.

```
\glsxtrdefaultsubsequentplfmt\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}
```

The default plural form for subsequent use of \texttt{\glsp}.

```
\Glsxtrdefaultsubsequentfmt\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}
```

The default sentence case singular form for subsequent use of \texttt{\Gls}.

```
\Glsxtrdefaultsubsequentplfmt\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}
```

The default sentence case plural form for subsequent use of \texttt{\Glsp}.

```
\GLSxtrdefaultsubsequentfmt\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}
```

The default all caps singular form for subsequent use of \texttt{\GLS}.

```
\GLSxtrdefaultsubsequentplfmt\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}
```

The default all caps plural form for subsequent use of \texttt{\GLSp}.
4. Abbreviations

4.5.3.2.4. Inline Full Format
These commands only need to be provided if the inline full form is different from the
display full form.

\glsxtrinlinefullformat{(entry-label)}{⟨insert⟩}
The singular full form of \glsxtrfull.

\glsxtrinlinefullplformat{(entry-label)}{⟨insert⟩}
The plural full form of \glsxtrfullpl.

\Glsxtrinlinefullformat{(entry-label)}{⟨insert⟩}
The sentence case singular full form of \Glsxtrfull.

\Glsxtrinlinefullplformat{(entry-label)}{⟨insert⟩}
The sentence case plural full form of \Glsxtrfullpl.

\GLSxtrinlinefullformat{(entry-label)}{⟨insert⟩}
The all caps singular full form of \GLSxtrfull.

\GLSxtrinlinefullplformat{(entry-label)}{⟨insert⟩}
The all caps plural full form of \GLSxtrfullpl.

4.5.3.2.5. Wrapper Commands
These are commands that can be used in the definitions of the above to ensure that the
appropriate accessibility fields and inner formatting is supported.

\glsxtrlongformat{(entry-label)}{⟨insert⟩}{⟨fmt-cs⟩}
This command is used in the definition of \glsxtrlong in some of the predefined abbreviation
styles to format the \texttt{long} value of the entry identified by \texttt{(entry-label)} with the command
\texttt{⟨fmt-cs⟩}, which should take one argument.
4. Abbreviations

Accessibility support is implemented with \glsaccesslong if the markwords attribute is true otherwise with \glsaccessfmtlong using \glsxtrgenentrytextfmt for the inner formatting.

This is then encapsulated (including or excluding the \textlangle insert\textrangle, according to \ifglsxtrinsertinside) with \textlangle fmt-cs\textrangle. If the \textlangle insert\textrangle content needs to be placed outside of \textlangle fmt-cs\textrangle, it will be individually encapsulated with the inner formatting.

\textbackslash Glsxtrlongformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\}

As above, but sentence case.

\textbackslash GLSxtrlongformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\}

As above, but all caps.

\textbackslash glsxtrlongplformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\}

As \textbackslash glsxtrlongformat, but for the longplural field.

\textbackslash Glsxtrlongplformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\}

As above, but sentence case.

\textbackslash GLSxtrlongplformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\}

As above, but all caps.

\textbackslash glsxtrlongformatgrp\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\}

As \textbackslash glsxtrlongformat, but adds grouping around \langle insert\rangle (with the inner formatting inside the group).

\textbackslash Glsxtrlongformatgrp\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\}
4. Abbreviations

As above, but sentence case.

`\GLSxtrlongformatgrp{(entry-label)}{(insert)}{(fmt-cs)}`

As above, but all caps.

`\glsxtrlongplformatgrp{(entry-label)}{(insert)}{(fmt-cs)}`

As \glsxtrlongplformat, but adds grouping around \langle insert \rangle (with the inner formatting inside the group).

`\Glsxtrlongplformatgrp{(entry-label)}{(insert)}{(fmt-cs)}`

As above, but sentence case.

`\GLSxtrlongplformatgrp{(entry-label)}{(insert)}{(fmt-cs)}`

As above, but all caps.

`\glsxtrshortformat{(entry-label)}{(insert)}{(fmt-cs)}`

This command is used in the definition of \glsxtrshort and in some of the predefined abbreviation styles to format the \texttt{short} value of the entry identified by \langle entry-label \rangle with the command \langle fmt-cs \rangle, which should take one argument.

Accessibility support is implemented with \glsaccessshort if the \texttt{markshortwords} attribute is true otherwise with \glsaccessfmtshort using \glsxtrgenentrytextfmt for the inner formatting.

This is then encapsulated (including or excluding the \langle insert \rangle, according to \ifglsxtr-insertinside) with \langle fmt-cs \rangle. If the \langle insert \rangle content needs to be placed outside of \langle fmt-cs \rangle, it will be individually encapsulated with the inner formatting.

`\Glsxtrshortformat{(entry-label)}{(insert)}{(fmt-cs)}`

As above, but sentence case.

`\GLSxtrshortformat{(entry-label)}{(insert)}{(fmt-cs)}`

177
As above, but all caps.

\glsxtrshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}

As \glsxtrshortformat, but for the \texttt{shortplural} field.

\Glsxtrshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}

As above, but sentence case.

\GLSxtrshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}

As above, but all caps.

\glsxtrshortformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}

As \glsxtrshortformat, but adds grouping around ⟨insert⟩ (with the inner formatting inside the group).

\Glsxtrshortformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}

As above, but sentence case.

\GLSxtrshortformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}

As above, but all caps.

\glsxtrshortplformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}

As \glsxtrshortformat, but adds grouping around ⟨insert⟩ (with the inner formatting inside the group).

\Glsxtrshortplformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}
As above, but sentence case.

\GLSxtrshortplformatgrp{(entry-label)}{(insert)}{(fmt-cs)}

As above, but all caps.

\glsxtrlongshortformat{(entry-label)}{(insert)}{(long-fmt-cs)}{(short-fmt-cs)}

A shortcut designed for \langle long \rangle (\langle short \rangle) styles. This is defined as:

\glsxtrlongformat{(entry-label)}{(insert)}{(long-fmt-cs)}\glsxtrfullsep{(entry-label)}\glsxtrparen{(\glsxtrshortformat{(entry-label)}{(short-fmt-cs)})}

Note that the \langle insert \rangle is only placed after the long form.

\Glsxtrlongshortformat{(entry-label)}{(insert)}{(long-fmt-cs)}{(short-fmt-cs)}

As above, but sentence case.

\GLSxtrlongshortformat{(entry-label)}{(insert)}{(long-fmt-cs)}{(short-fmt-cs)}

As above, but all caps.

\glsxtrlongshortplformat{(entry-label)}{(insert)}{(long-fmt-cs)}{(short-fmt-cs)}

As \glsxtrlongshortformat but uses the plural versions \glsxtrlongplformat and \glsxtrshortplformat.

\Glsxtrlongshortplformat{(entry-label)}{(insert)}{(long-fmt-cs)}{(short-fmt-cs)}
4. Abbreviations

As above, but sentence case.

\GLSxtrlongshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As above, but all caps.

\glsxtrshortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

A shortcut designed for (short) (long) styles. This is defined as:

\glsxtrshortformat{⟨entry-label⟩}{⟨insert⟩}{⟨short-fmt-cs⟩}\glsxtrfullsep{⟨entry-label⟩}\glsxtrparen{\glsxtrlongformat{⟨entry-label⟩}{⟨long-fmt-cs⟩}}

Note that the ⟨insert⟩ is only placed after the short form.

The syntax is the same as for \glsxtrlongshortformat even though \glsxtrlongformat and \glsxtrshortformat are flipped within the definition.

\Glsxtrshortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As above, but sentence case.

\GLSxtrshortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As above, but all caps.

\glsxtrshortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}

As \glsxtrshortlongformat but uses the plural versions \glsxtrshortplformat and \glsxtrlongplformat.
4. Abbreviations

As above, but sentence case.

\GLSxtrshortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-CS⟩}{⟨short-fmt-CS⟩}

As above, but all caps.

4.6. Restoring Base Acronym Mechanism

It’s possible to revert \newacronym back to the definition provided by the base glossaries package. However, if you do this, you will lose all the abbreviation features provided by glossaries-extra.

Where possible, avoid this. If you’re simply trying to make the long form appear on first use with \newacronym, set the abbreviation style using:

\setabbreviationstyle[acronym]{long-short}

If you really need to use the original base glossaries package’s acronym mechanism, it’s better to stick with just glossaries and not use glossaries-extra. However, it may be that you need to use a glossaries-extra feature, such as \printunsrtglossary, but you have a custom acronym style that you can’t implement using the glossaries-extra abbreviation mechanism. This is a rare edge case for unusual formats, as it should be possible to implement most common abbreviation formats using the predefined styles.

Unpredictable results will occur if \RestoreAcronyms or \MakeAcronymsAbbreviations are used after abbreviations or acronyms have been defined.

\RestoreAcronyms

Restores \newacronym back to the original base glossaries interface. Note that this doesn’t affect \newabbreviation. It also sets the regular attribute for the acronym category to false and sets the acronym style to long-short (which is the default for the base package).

The display style for each glossary identified in the acronym lists is switched to the default
4. Abbreviations

acronym display style.

\MakeAcronymsAbbreviations

Counteracts \RestoreAcronyms.
Entries can be referenced using the \gls-like and \glstext-like commands, as described in the base glossaries manual. There are some additional commands provided by glossaries-extra:

- abbreviation commands, such as \glsxtrshort (see §4);
- commands for use in captions or section headings, such as \glsfmttext (see §5.3.2);
- commands, such as \glsxtrp, designed for use within fields to help mitigate the problems of nesting (see §5.4);
- commands, such as \mglsl, that are designed for referencing multi (or compound) entries (see §7);
- commands, such as \glsaccessname, used to incorporate accessibility support (see §9.2);
- commands, such as \dgls, that are designed for \bibtex\gls's dual entries (see §11.5.7);
- commands, such as \rglsl, that depend on the number of entry records (see §11.4).

Additionally, the entry counting commands, such as \cglsl, provided by the base glossaries package are modified by glossaries-extra (see §6.1).

The \gls-like commands are designed to produce text at that point in the document (the link text, §5.5), index the entry (to ensure that it appears in the glossary, §5.8) and unset the first use flag (which can alter the link text, §5.10). Additional information can be appended automatically with the post-link hook (§5.5.4). The link text is given by the entry style (see §5.5.5) or by the final argument of \glsdisp.

The \glstext-like commands are designed to produce text at that point in the document (the link text, §5.5) and index the entry (to ensure that it appears in the glossary, §5.8). Additional information can be appended automatically with the post-link hook (§5.5.4). The link text is determined by the calling command. For example, the corresponding field value (possibly encapsulated with \glsxtrregularfont and the inner formatting) for commands like \glstext or the final argument of \glslink.

The \gls-like and \glstext-like commands can all be used with a star (*) or plus (+) modifier. The star modifier automatically implements \texttt{hyper=false} (disables the hyperlink) and the plus modifier automatically implements \texttt{hyper=true} (forces the hyperlink on, if supported).
With glossaries-extra, it’s possible to define an additional modifier for your own use with:

\GlsXtrSetAltModifier{⟨token⟩}{⟨options⟩}

The ⟨token⟩ must be a single token, so a multi-byte UTF-8 character will required a native Unicode engine (XƎLATEX or LuaLATEX). For example, the following:

\GlsXtrSetAltModifier{!}{format=glsignore}

means that \gls{⟨label⟩} will be equivalent to \gls[format=glsignore]{⟨label⟩}.

It’s also possible to redefine the star and plus modifiers:

\GlsXtrSetStarModifier{⟨options⟩}

This sets the options to use for the star modifier.

\GlsXtrSetPlusModifier{⟨options⟩}

This sets the options to use for the plus modifier. For example, the following:

\GlsXtrSetPlusModifier{noindex}

means that the star modifier will now suppress indexing instead of switching on the hyperlink.

The \gls-like and \glstext-like commands have a complicated internal structure, which can be viewed as a series of layers. The outermost common layer is:

\% save settings
\% initialise options, see §5.1
\glslinkwrcontent{⟨index & fmt content⟩}
\% restore settings
\% post-link hook, see §5.5.4

The ⟨index & fmt content⟩ consists of the indexing (see §5.8) and the (possibly hyperlinked)
5. Referencing (Using) Entries

formatted text, see §5.5. The \textit{\langle index \& fmt content\rangle} code is encapsulated with:

\begin{verbatim}
\glslinkwrcontent{(code)}
\end{verbatim}

In v1.48, this was added to scope the link text and indexing code, which helped to prevent unwanted spacing caused by the whatsit and also helped to prevent some setting leakage, in the event of nesting (see §5.4), but this caused spacing issues when used in math mode, so from v1.49 this command now simply does its argument. The whatsit is now scoped with \glsencapwrcontent instead.

The \glssxtrp command, designed for nested use, deals with the problem by suppressing the post-link hook and adding an outer group. For example, \glssxtrp{short}{html} behaves like:

\begin{verbatim}
{\let\glspostlinkhook\relax
 \glssxtrshort[noindex, nohyper]{html}}
\end{verbatim}

Depending on the settings (the wrgloss option or the wrgloss attribute), the indexing may come before the text:

\begin{verbatim}
\glslinkwrcontent{(index) (fmt content)}
\end{verbatim}

or after the text:

\begin{verbatim}
\glslinkwrcontent{(fmt content) (index)}
\end{verbatim}

or may be suppressed with noindex:

\begin{verbatim}
\glslinkwrcontent{(fmt content)}
\end{verbatim}

The \textit{\langle fmt content\rangle} part is described in §5.5. The \textit{\langle index\rangle} part is the actual indexing (see §5.8) but also increments the index count, if applicable. Both the associated whatsit and increment are encapsulated with \glsencapwrcontent.

Avoid using \texttt{\glsstext}, \texttt{\glsplural}, \texttt{\glsfirst} and \texttt{\glsfirstplural} (and their case-changing variants) with entries that have been defined with \texttt{\newabbreviation}. Some of the abbreviation styles are too complicated to work with those commands. Instead, use commands like \texttt{\glssxtrshort}, \texttt{\glssxtrfull} or use \texttt{\gls} with the \texttt{prefix} or \texttt{unprefix} options.

The base glossaries package provides a way to adjust the formatting of the link text for the \texttt{\gls}-like commands according to the glossary type with \texttt{\defglsentryfmt}. The glossaries \texttt{-extra} package changes the default entry formatting (§5.5.5) and provides additional ways of
modifying the displayed content (§5.5).

The heading commands (described in §5.3) are designed to prevent indexing or changes to the first use flag if they appear in the table of contents (or list of figures, etc) or if they appear in the page header.

Although the base glossaries package warns against nested link text, the glossaries-extra package provides \glsxtrp which can be used instead of \gls in field values to overcome some of the associated problems. See §5.4 for further details.

If you need to simply access a field value without any formatting, see §5.11. (See §3.5 to set field values.) If you want to encapsulate the value with the appropriate accessibility tag, see §9.2.

Commands such as \glsadd (see §5.8) and \glssee (see §5.9) are designed to only index (to ensure the entry appears in the glossary) without producing any text or changing the first use flag.

The \gls-like, \glstext-like and \glsadd commands all have an initial optional argument that can be used to override the default actions. Some options are only applicable for particular subsets of referencing commands. For example, noindex is pointless for \glsadd since the sole purpose of that command is to index. Whereas types is only available with \glsaddall.

5.1. Options

Some options are provided by the base glossaries package, but there are some additional options provided by glossaries-extra, which are listed in §5.1.2. Below, ⟨option-list⟩ indicates the options that are passed in the optional argument of the calling command (such as \gls).

The order that the options are applied is:

1. prereset, preunset and postunset options are initialised by \glsinitreunsets;
2. hyper is initialised by \glsxtrchecknohyperfirst (\glsfirst-like only);
3. wrgloss option is initialised by \glsxtrinitwrgloss (not implemented by \glsadd or \glsxtrfmt);
4. hyperoutside option is initialised by \glsxtrinithyperoutside (not implemented by \glsadd or \glsxtrfmt);
5. initialise noindex=false (not \glsadd);
6. options identified by \GlsXtrSetDefaultValue\glsOpts, \GlsXtrAppToDefaultValue\glsOpts or \GlsXtrPreToDefaultValue\glsOpts (not implemented by \glsadd);
7. (\glsxtrfmt only) options provided in \GlsXtrFmtDefaultValueOptions;
8. (\gls-like only) the hyperfirst package option, nohyperfirst attribute and nohypernext attributes are checked to determine if the hyper option should be switched off (tests followed by \glslinkcheckfirsthyperhook);
5. Referencing (Using) Entries

9. \glslinkpresetkeys (not implemented by \glsadd or \glsxtrfmt);
10. (\glsadd only) \glsaddpresetkeys;
11. \langle option-list \rangle;
12. \glslinkpostsetkeys (provided by the base glossaries package, not implemented by \glsadd or \glsxtrfmt);
13. (\glsadd only) \glsaddpostsetkeys.

5.1.1. Setting Up Defaults

You can (locally) set your preferred default options for the \gls-like and \glstext-like commands using:

\GlsXtrSetDefaultGlsOpts{\langle options \rangle}

The \langle options \rangle may be any options that you can pass to those commands. These options also apply to \glsxtrfmt but not to \glsadd.

Note that multiple instances of \GlsXtrSetDefaultGlsOpts will override each other.

If you want to add to the existing options, you can use one of the following commands (both may be scoped).

\GlsXtrAppToDefaultGlsOpts{\langle options \rangle}

Appends \langle options \rangle to the list of default options.

\GlsXtrPreToDefaultGlsOpts{\langle options \rangle}

Prepends \langle options \rangle to the list of default options.

For example, to prevent indexing in the front matter and back matter but not in the main matter:

\frontmatter
\GlsXtrSetDefaultGlsOpts{noindex}
...
\mainmatter
Note that `noindex=false` is now set before the options given in `\GlsXtrSetDefaultGlsOpts` to ensure that the setting is correctly initialised, so as from v1.49 you can simply set an empty options list to reset the default. Prior to v1.49, it was necessary to ensure that the `noindex` key was always present in the options list to avoid instability. So for pre v1.49, the line after `\mainmatter` in the above would need to be:

```
\GlsXtrSetDefaultGlsOpts{noindex=false}
```

The default location encap is `glsnumberformat` but can be changed (locally) with:

```
\GlsXtrSetDefaultNumberFormat{(encap)}
```

This can be overridden by explicitly setting the `format` key.

The default options for `\glsxtrfmt` only are given by:

```
\GlsXtrFmtDefaultOptions
```

This command should simply expand to the required list of options. These options are set after any options given in `\GlsXtrSetDefaultGlsOpts` and before `<option-list>`.

```
\glslinkpresetkeys
```

This hook is performed after any settings provided in `\GlsXtrSetDefaultGlsOpts` but before `<option-list>`. This hook also applies to `\glsxtrfmt` but not to `\glsadd`.

Note that `\glslinkpostsetkeys`, provided by the base glossaries package, is performed after `<option-list>` is processed.

```
\glsaddpresetkeys
```

This hook, which is only used by `\glsadd`, is performed before `<option-list>`.

```
\glsaddpostsetkeys
```

188
5. Referencing (Using) Entries

This hook, which is only used by \glsadd, is performed after \emph{(option-list)}.

\texttt{\glsinitreunsets}

This hook initialises the pre unset/reset options to: \texttt{prereset=none} and \texttt{preunset=none}. It also initialises the \texttt{postunset} setting to perform the post-unset (where applicable) but it will retain the current local/global setting.

This hook will also implement the local repeat unset feature of \texttt{\GlsXtrUnsetBuffer-EnableRepeatLocal}.

\texttt{\glsxtrchecknohyperfirst{⟨entry-label⟩}}

This hook is only used by \glsfirst, \glsfirstplural and their case-changing variants. The hook will implement \texttt{hyper=false} if the \texttt{nohyperfirst} attribute is set to \texttt{true}.

\texttt{\glsxtrinitwrgloss}

This hook initialises the default setting of the \texttt{wrgloss} option. If the \texttt{wrgloss} attribute is set to \texttt{after} then this implements \texttt{wrgloss=after} otherwise it implements \texttt{wrgloss=before}. This setting can subsequently be overridden by \texttt{\GlsXtrSetDefaultGlsOpts}, \texttt{\glslinkpresetkeys}, the \emph{(option-list)} argument or \texttt{\glslinkpostsetkeys}. This hook also applies to \texttt{\glsxtrfmt} but not to \texttt{\glsadd}.

If you prefer to have the default to place the indexing after the link text, you can redefine this hook as follows:

\begin{verbatim}
\renewcommand*{\glsxtrinitwrgloss}{%
  \glsifattribute{\glslabel}{wrgloss}{before}%
    \glsxtrinitwrglossbeforetrue%
    %
    \glsxtrinitwrglossbeforefalse%
  %}
\end{verbatim}

\texttt{\glsxtrinithyperoutside}

This hook initialises the default setting of the \texttt{hyperoutside} option. If the \texttt{hyperoutside}
attribute is set to false then this implements \hyperoutside=false otherwise it implements \hyperoutside=true. This setting can subsequently be overridden by \GlsXtrSetDefaultGlsOpts, \glslinkpresetkeys, the \glslinkpostsetkeys. This hook also applies to \glsxtrfmt but not to \glsadd.

Within any of the hooks that are used by the \gls-like, \glstext-like or \glsxtrfmt commands, you can set options using:

\setupglslink{\langle options\rangle}

Within any of the hooks that are used by \glsadd, you can set options with:

\setupglsadd{\langle options\rangle}

### 5.1.2. Additional Options

Options for the \gls-like and \glstext-like commands that are provided by the base glossaries package also apply to new commands like \glsxtrfmt and \glstextfmt. In addition, the options below are provided by glossaries-extra. Note that some options, such as \postunset, only apply to the \gls-like commands. Options that relate to the hyperlink, formatting, first use flag or whether/where (noindex/wrgloss) to perform indexing aren’t available for \glsadd.

\hyperoutside=\langle boolean\rangle  \hspace{1cm} default: \text{true}; \hspace{0.2cm} initial: \text{true}

This boolean option determines whether the hyperlink should be inside or outside of \glstextformat (see §5.5.1). If true, the link text is encapsulated as:

\langle\hyperlink-cs\rangle\{\langle target\rangle\}\{\glstextformat\{\langle text\rangle\}\}

otherwise it’s encapsulated as:

\glstextformat\{\langle\hyperlink-cs\rangle\{\langle target\rangle\}\{\langle text\rangle\}\}

where \langle\hyperlink-cs\rangle is the command that generates the hyperlink (if enabled).

\textformat=\langle csname\rangle
5. Referencing (Using) Entries

The value of this key should be the name of a control sequence (without the leading backslash). If this option is set, the given control sequence will be used instead of \glistextformat to encapsulate the link text. Note that this control sequence should take a single argument (the link text). See §5.5.1 for further details.

This option will override the textformat attribute.

\innertextformat=⟨csname⟩ initial: glsxtrefformat

The value of this key should be the name of a control sequence (without the leading backslash). The command \glsxtrgenentrytextfmt (which shouldn’t be redefined) is assigned to this control sequence at the start of the \gls-like and \glistext-like commands. This command is used within the predefined abbreviation styles and within \glsgenentryfmt to encapsulate the entry field values.

Custom styles that don’t use \glsxtrgenentrytextfmt won’t support this key. See §5.5.3 for further details.

Some formatting commands require direct access to the actual text or else the content has to be placed inside a box (which inhibits line-breaking). These commands won’t work with textformat as the text is usually too deeply embedded. This option provides a way of using those problematic commands, however there’s still no guarantee that they will work (for example, in the case of custom styles or where the field value itself contains commands).

postunset=⟨value⟩ default: global; initial: global

This option only applies to the \gls-like commands and indicates whether or not to unset the first use flag after the link text. It may take one of three values: global (behaves like local=false), local (behaves like local=true) or none (doesn’t unset the first use flag after the first use). See §5.10.

prerest=⟨value⟩ default: local; initial: none

This option may take one of three values: none (no reset), local or global. This option (if not none) will reset the first use flag before the link text and additionally change \glsxtrifwasfirstuse so that it indicates that this was the first use of the entry. See §5.10.

Note that this is different from using \glslocalreset or \glsreset before the \glistext-like commands. Normally \glistext and \glspplural will define \glsxtrifwasfirstuse
so that it indicates that this was not the first use of the entry (regardless of whether or not the entry has actually been used).

For example:

```latex
\glsdefpostlink{general}{\glsxtrpostlinkAddDescOnFirstUse}
\newglossaryentry{sample}{name={sample},
  first={sample first use},description={an example}}
\begin{document}
\text field: \glstext{sample}.
First use: \gls{sample}. Next use: \gls{sample}.
Force reset: \gls[prereset]{sample}.
Used? \ifglsused{sample}{Yes}{No}.
Force reset: \glstext[prereset]{sample}.
Used? \ifglsused{sample}{Yes}{No}.
\end{document}
```

Example 90: Illustrating the \texttt{prereset} option

Text field: sample.
First use: sample first use (an example). Next use: sample.
Force reset: sample first use (an example). Used? Yes.

Note that \texttt{\gls} unsets the first use flag (unless \texttt{postunset=none}), so the sample entry is marked as used afterwards, but \texttt{\glstext} doesn’t alter the first use flag, after the link text so the sample entry is still marked as unused afterwards.

```
\texttt{\texttt{preunset}=(value)} \hspace{1cm} \texttt{default: local; initial: none}
```

This option may take one of three values: none (no unset), local or global. This option (if not none) will unset the first use flag before the link text and additionally change \texttt{\glsxtr-ifwasfirstuse} so that it indicates that this wasn’t the first use of the entry. See §5.10.

Note the effect of using a global reset but a local unset in the example below. Both options are performed, but the unset locally overrides the global reset.
5. Referencing (Using) Entries

```latex
\glsdefpostlink{general}{\glsxtrpostlinkAddDescOnFirstUse}
\newglossaryentry{sample}{name={sample},
  first={sample first use},description={an example}}
\begin{document}
\gls{sample}. Used? \ifglsused{sample}{Yes}{No}.

{\glsfirst[preunset=local,prereset=global]{sample}.
  Used? \ifglsused{sample}{Yes}{No}.
}

Used? \ifglsused{sample}{Yes}{No}.

{\gls[preunset=local,prereset=global]{sample}.
  Used? \ifglsused{sample}{Yes}{No}.
}

Used? \ifglsused{sample}{Yes}{No}.
\end{document}
```

Example 91: Combining \texttt{prereset} and \texttt{preunset}

sample first use (an example). Used? Yes.

Used? No.

sample. Used? Yes.

Used? Yes.

Remember that \texttt{\gls} globally unsets the first use flag (unless changed with \texttt{postunset}), which counteracts \texttt{prereset=global}.

\textbf{noindex} = \langle boolean \rangle  \\
\textit{default:} true; \textit{initial:} false

This is a boolean option that determines whether or not to suppress the normal indexing. For example, to prevent any locations in the front matter or back matter appearing in the glossary:

```latex
\frontmatter
\GlsXtrSetDefaultGlsOpts{noindex}
...```

193
5. Referencing (Using) Entries

\mainmatter
\GlsXtrSetDefaultGlsOpts{noindex=false}
...
\backmatter
\GlsXtrSetDefaultGlsOpts{noindex}

Note that if you are using auto-indexing (see §12), noindex=false will also suppress the auto-indexing.

If you are using bib2gls, you may want to consider instead using format=glsignore to create an ignored location that ensures the entry is selected without adding a location to the location list. (Don’t use this method for the other indexing methods as you’ll end up with invisible locations with spurious commas in your location lists.)

\textit{wrgloss=\langle position\rangle}
\textit{initial: before}

This option may take one of two values, before or after, which indicate whether the indexing should occur before or after the link text. The indexing creates a whatsit that can interfere with spacing or cause other problems. The other thing to consider is where the link text is long, such as a phrase or full form of an abbreviation, that may be split by a page break. You will need to decide if you want the indexing before the link text, so that the location is at the end of the page where the text starts, or if you want the indexing after the link text, so that the location is at the start of the next page where the text ends.

This option corresponds to a conditional:

\texttt{\ifglsxtrinitwrglossbefore \langle true\rangle\else \langle false\rangle\fi}
\textit{initial: \iftrue}

The hook \glsxtrinitwrgloss sets this conditional according to whether or not the \textit{wr-}
\textit{gloss} attribute has been set to after:

\texttt{\newcommand*{\glsxtrinitwrgloss}{\%
\glsifattribute{\glslabel}{wrgloss}{after}\%
\{\glsxtrinitwrglossbeforefalse\%
\{\glsxtrinitwrglossbeforetrue\%
\}}

\textit{thevalue=\langle location\rangle}

Sets the entry location to the given value instead of obtaining it from the location counter. If you are using hyperref you may also need to set the location’s hypertarget with the\texttt{Hvalue}.
This option is primarily intended for use with `bib2gls` to supply locations that don’t have an associated counter within the document, such as an external location. If you want to automatically add locations from a supplemental document to an entry’s location list, you can use the `supplemental-locations` resource option. See the `bib2gls` user manual for further details.

For example, to index a location in a supplementary document:

```
\glsadd[thevalue={Suppl. \ 2.45}]{sample}
```

This will add “Suppl. 2.45” to the location list for the “sample” entry.

Note that the value must conform to the indexing application’s location syntax. For `makeindex`, this is limited to Roman, roman, arabic, alph and Alph. With `xindy`, the location syntax must be defined in the `xindy` module (standard location syntaxes are supplied by default). There’s no restriction on the location syntax for `bib2gls`, although if it can’t deduce a numerical value it won’t be able to form a range.

If you want a hyperlink to an external file, you can use:

```
\glsxtrsupphypernumber{⟨location⟩}
```

as the formatting command for the location encap. For example:

```
\glsadd[thevalue=S.2,format=glsxtrsupphypernumber]{sample}
```

The path to the external file needs to be set in the `externallocation` category attribute.

The hyperlink for the supplementary location may or may not take you to the relevant place in the external PDF file depending on your PDF viewer. Some may not support external links, and some may take you to the first page or last visited page.

For example, if both `sample-suppl-hyp.pdf` and `sample-suppl-main-hyp.pdf` are in the same directory, then viewing `sample-suppl-main-hyp.pdf` in Evince will take you to the correct location in the linked document (when you click on the S.2 external link), but Okular will take you to the top of the first page of the linked document.

This method can only be used where there is one external source for the designated cat-
5. Referencing (Using) Entries

egory (identified by the `externallocation` attribute). For multiple sources, you need to use `bib2gls` v1.7+, which is the better method in general as it can automatically fetch the relevant locations from the aux files of the designated external documents without the need to explicitly use `\glsadd`.

\[ \text{theHvalue} = \langle \text{the-H-value} \rangle \]

Sets the hypertarget corresponding to the location, which will be used if the `format` supports hyperlinks. This is analogous to `hyperref`'s `\the\langle \text{counter-name} \rangle` that provides the hypertarget for a reference to `\the\langle \text{counter-name} \rangle`.

This option is primarily intended for use with the `thevalue` option.

Unless you are using `record=nameref`, you must ensure that it’s possible to form `\langle \text{the-H-value} \rangle` from `\langle \text{h-prefix} \rangle \langle \text{thevalue} \rangle` for some `\langle \text{h-prefix} \rangle` (where `\langle \text{thevalue} \rangle` is given by `thevalue` or the value of the location counter). This restriction is due to the limitations imposed by `makeindex` and `xindy`.

\[ \text{prefix} = \langle \text{link-prefix} \rangle \]

This option locally redefines `\glolinkprefix` to `\langle \text{link-prefix} \rangle`. If you are using `\printunsrtglossary` to redisplay a list (possibly in a different order) then you will need some way of changing the entry targets to avoid duplicate hyperlink targets. One way of achieving this is to redefine `\glolinkprefix` for the subsequent lists. You will then need to use the `prefix` option in commands like `\gls` to ensure that the hyperlink for the link text points to the desired list.

This option is intended for use with the “unsrt” family of commands and `\glsextracopytoglossary` (which is used by `bib2gls`). The other indexing methods don’t support repeated lists.

5.2. Case Changing

Case-changing commands, such as `\Gls` and `\GLS`, perform the conversion using commands provided by `mfirstuc`. The underlying commands provided by `mfirstuc` were redesigned in `v2.08` to use the newer, better case-changing commands available with the `BfPpX3` kernel. The base `glossaries` package `v4.50` and `glossaries-extra` `v1.49` were developed concurrently with
mfirstuc v2.08 to take advantage of the new features. Version 1.49 of glossaries-extra was also developed concurrently with bib2gls v3.0 which, in turn, was developed alongside version 0.9.2.7b of the \TeX\ parser library.

It’s not possible to upload all these new versions at the same time, so it will be necessary to stagger their deployment. The new case-changing features will work best when all these new versions are installed. In the interim, a reduced feature set will be used.

### 5.2.1. Sentence Case Commands

Both the base glossaries package and the glossaries-extra package provide sentence case commands, which convert the first letter to uppercase. These are provided for situations where an entry is referenced at the start of a sentence. Sentence-casing is also implemented when the attributes `glossname` or `glossdesc` are set to `firstuc`.

The case conversion is performed using:

\[
\texttt{\textbackslash glssentencecase\{⟨text⟩\}}
\]

The default definition uses `\makefirstuc`, which is provided by the mfirstuc package. This was originally part of the base glossaries package, but was split into a separately distributed package in 2015. Back then, there was no expandable sentence-case command. There was also a problem with referencing entries where link text was encapsulated with a text-block command (which occurs, in particular, with acronym and abbreviation styles). The first letter of the text-block command’s argument needed to be obtained, which resulted in some trickery that proved problematic with UTF-8.

The \Eplain\X kernel now provides a suitable expandable command that works with UTF-8, and mfirstuc v2.08+ provides `\MFUsentencecase` that directly interfaces with it. If an older version of mfirstuc is installed, glossaries v4.50+ and glossaries-extra v1.49+ will provide `\MFUsentencecase`. You can use this in expandable contents. For example:

\[
\texttt{\section{\MFUsentencecase\{\glsentrytext{label}\}}}
\]

However, in the above example, it’s simpler to do:

\[
\texttt{\section{\Glsfmttext{label}}}
\]

If hyperref has been loaded, `\Glsfmttext{label}` will now expand to:

\[
\MFUsentencecase\{\glsentrytext{label}\}
\]

in the PDF bookmark.
Internally, \makefirstuc now uses \MFUsentencecase to perform the case conversion, but it still parses its argument to determine if it starts with \textlangle cs\rangle\{\langle text\rangle\}. This means that with mfirstuc v2.08+, you now don’t have to worry about UTF-8 characters occurring at the start of the text.

For example, with mfirstuc v2.07 you would need to do something like:

\begin{verbatim}
\newglossaryentry{elite}{name={{é}lite},
description={...}}
\end{verbatim}
in order for \textGls{elite} to work. Whereas with mfirstuc v2.08, you can now simply do:

\begin{verbatim}
\newglossaryentry{elite}{name={élite},
description={...}}
\end{verbatim}

(As from glossaries v4.47, it should be possible to use UTF-8 characters in the label as well.)

Whilst you can redefine \textglsentencecase to use \MFUsentencecase directly (without using \makefirstuc as an intermediary), this may result in content being expanded that wouldn’t have been expanded previously. In particularly, if \textlangle cs\rangle isn’t robust and expands to content that includes labels then the case-change can fail. You also won’t be able to take advantage of the blockers and mappings that are only recognised as such by \makefirstuc. If you use \MFUsentencecase instead, blockers and mappings will be treated as exclusions, which are likely to result in unwanted side-effects.

Both \makefirstuc and \MFUsentencecase recognise exclusions. These are text-block commands which take a single mandatory argument that needs to be skipped. For example, in the following \textglsadd{example} needs to be skipped:

\begin{verbatim}
\MFUsentencecase{\glsadd{example}some text}
\end{verbatim}

Exclusions are identified with \textMFUexcl. If you have an older version of mfirstuc, this won’t be defined, so glossaries v4.50+ and glossaries-extra v1.49+ provide:

\begin{verbatim}
\glsmfuexcl{\langle cs\rangle}
\end{verbatim}

This will use \textMFUexcl with mfirstuc v2.08+. With older versions, a definition will be provided that works with \MFUsentencecase, but exclusions won’t be recognised by \makefirstuc.

As from glossaries v4.50, \textglsadd will be identified as an exclusion (via \textglsmfuexcl), but the optional argument will cause a problem if present. See the mfirstuc v2.08+ manual for a workaround. Note that commands such as \textglsaddall and \textglsaddeach aren’t identified as exclusions as they aren’t expected to occur in text that may require a case-change.
5. Referencing (Using) Entries

With glossary entry references, there are commands that take a label as the argument, which shouldn’t have any case-changed applied, but also shouldn’t be skipped. For example:

\makefirstuc{\GLS{example} something}

In this situation, there shouldn’t be any case-change as \GLS already implements a case-change. This type of command is referred to as a blocker in the \mfirstuc manual, as it indicates a command that should prevent any case-change if it’s encountered at the start of the text. Blockers are identified with \MFUblocker. If you have an older version of \mfirstuc, this won’t be defined, so glossaries v4.50+ and glossaries-extra v1.49+ provide:

\glsmfublocker{(cs)}

This will use \MFUblocker with \mfirstuc v2.08+. With older versions, it will simply use \gls-mfuexcl which will instead identify the command as an exclusion and won’t be recognised by \makefirstuc. See the \mfirstuc v2.08+ manual for further information about blockers.

As from glossaries v4.50+, commands like \GLS will be identified as blockers using \glsmfublocker, and glossaries-extra now identifies similar commands, such as \rGLS as blockers.

Finally, there are mappings. These are commands that should be substituted with another command, which is expected to perform the case-change. For example:

\makefirstuc{\gls{example} something}

This shouldn’t skip or block \gls but instead should convert the text to:

\Gls{example} something

This is implemented by adding a mapping from \gls to \Gls. Mappings are added using \MFUaddmap. If you have an older version of \mfirstuc, this won’t be defined, so glossaries v4.50+ and glossaries-extra v1.49+ provide:

\glsmfuaddmap{(cs1)}{(cs2)}

This will use \MFUaddmap with \mfirstuc v2.08+. With older versions, it will simply use \gls-mfuexcl which will instead identify the command as an exclusion and won’t be recognised by \makefirstuc. See the \mfirstuc v2.08+ manual for further information about mappings.

As from glossaries v4.50+, commands like \gls will be mapped to the appropriate sentence case command using \glsmfuaddmap, and glossaries-extra now identifies similar mappings,
such as \rfr{rgls} mapped to \rfr{RGLS}.

In order to integrate the full set of features provided by mfirstuc v2.08+, you will need both glossaries v4.50+ and glossaries-extra v1.49+.

5.2.2. Lower Case

\gllowercase{(text)}

This is defined by glossaries v4.50+ to use the \texttt{\LaTeX3} command to convert to lowercase. If an older version of glossaries is present, then this command will be provided by glossaries-extra but it will be defined to use \MakeTextLowercase instead. This command is primarily provided for use with small caps styles to convert an abbreviation to lowercase, but isn’t actually used anywhere by default.

5.2.3. Upper Case

\glsuppercase{(text)}

This is defined by glossaries v4.50+ to use the \texttt{\LaTeX3} command to convert to uppercase (all caps). If an older version of glossaries is present, then this command will be provided by glossaries-extra but it will be defined to just use \texttt{mfirstuc\MakeUppercase}, which is provided by mfirstuc. This command is used by all caps commands such as \texttt{\GLSxtrusefield}.

5.2.4. Title Case

\glscapitalisewords{(content)}

This is defined by glossaries v4.48 to use \texttt{\capitalisewords} to convert to title case. If you experience any errors with title case commands, such as \glsentrytitlecase, or attributes such as \texttt{glossdesc} then try redefining this command to use \texttt{\capitalisefmtwords*} instead. See the mfirstuc manual for further details.

5.3. Entries in Sectioning Titles, Headers, Captions and Contents

The glossaries user manual cautions against using commands like \gls in chapter or section titles. The principle problems are:

- if you have a table of contents, the first use flag will be unset in the contents rather than later in the document;
5. Referencing (Using) Entries

- if you have the location lists displayed in the glossary, unwanted locations will be added to it corresponding to the table of contents (if present) and every page that contains the entry in the page header (if the page style in use adds the chapter or section title to the header);

- if the page style in use adds the chapter or section title to the header and attempts to convert it to uppercase, the entry label (in the argument of \gls etc) will be converted to uppercase and the entry won’t be recognised;

- if you use hyperref, commands like \gls can’t be expanded to a simple string and only the label will appear in the PDF bookmark (with a warning from hyperref);

- if you use hyperref, you will end up with nested hyperlinks in the table of contents.

Similar problems can also occur with captions (except for the page header and bookmark issues).

The glossaries-extra package tries to resolve the header problem by modifying \markright, \markboth and \@starttoc. If this causes unwanted side-effects, you can restore their former definitions using:

\glsxtrRevertMarks

This will revert \markright, \markboth and \@starttoc back to the definitions in effect when glossaries-extra was loaded. Alternatively, you can use:

\glsxtrRevertTocMarks

This will only revert \@starttoc.

If you use \glsxtrRevertMarks or \glsxtrRevertTocMarks, you will need to employ the simplistic approach, described in §5.3.1, which is the method recommended by the glossaries user manual. Otherwise, you can use the commands described in §5.3.2, which provide a better solution.

5.3.1. Simplistic Approach

To get around all these problems, the glossaries user manual recommends using the expandable non-hyperlink commands, such as \glsentrytext (for regular entries) or \glsentryshort (for abbreviations). This is the simplest solution, but doesn’t allow for special formatting that’s applied to the entry through commands like \glstext or \glsxtrshort.

For example:
\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\glsdefpostlink{general}{\glsxtrpostlinkAddDescOnFirstUse}
\newglossaryentry{sample}{name={sample},description={an example}}
\newglossaryentry{alpha}{name={\ensuremath{\alpha}},description={\alpha}}
\begin{document}
\tableofcontents
\section{\Glsentrytext{sample} and \glsentrytext{alpha}}
First use: \gls{sample} and \gls{alpha}.
Next use: \gls{sample} and \gls{alpha}.
\printunsrtglossary
\end{document}
Example 92: References in section headings (simplistic approach)

Contents

1 Sample and α 1

Glossary

1 Sample and α

First use: sample (an example) and α (alpha).

Next use: sample and α.

Glossary

sample an example

α alpha

This solves some problems: it avoids nested links in the table of contents, the first use flag isn’t prematurely unset and the PDF bookmarks has a reasonable substitution, but it still isn’t a complete solution as the above document will fail if the page style is changed to headings and a page break is inserted before the section (after \tableofcontents), which will lead to the error:

Glossary entry `SAMPLE' has not been defined.

This is because the case-change applied to the header converts the label “sample” to “SAMPLE”, which doesn’t correspond to a defined entry. (This can now be avoided with mfirstuc v2.08+.)

If the case conversion is applied by, then the case-change can be prevented by encapsulating the label with \NoCaseChange, but this ends up quite complicated. This is actually what the commands describe in §5.3.2 do when they are in a heading. This allows for older versions of mfirstuc that don’t recognise exclusions. See §5.2 for further details.

The \NoCaseChange command was originally provided by the textcase package to prevent \MakeTextUppercase from applying a case-change. The functionality of the textcase package has now been absorbed into the \LaTeX kernel, which means that as from 2022, textcase is deprecated and \NoCaseChange is defined by the kernel.
5. Referencing (Using) Entries

5.3.2. New Commands Designed for Chapter/Section Headings or Captions

This section is irrelevant if you use \glsxtrRevertMarks to restore the definitions of \markright, \markboth and \@starttoc. If you use \glsxtrRevertTocMarks, then this section is only applicable to \markright and \markboth.

The commands listed here are designed for use within captions or section headings. They are designed to overcome some of the problems illustrated in the previous section. Note that they only have a single argument, the entry label. There are no optional arguments. Below, “header” refers to page header text added with \markright or \markboth, and “contents” refers to the table of contents or any other “list of” that uses \@starttoc, such as the list of figures.

Each command \glsfmt⟨field⟩ (such as \glsfmttext or \glsfmtshort) behaves like an analogous \gls⟨field⟩ or \glsxtr⟨field⟩ command (such as \glstext or \glsxtrshort) but with the options noindex and hyper=false and no insert. When they occur within a header, they are protected from having any case-change applied (which will interfere with the entry label). Since this means they won’t appear in all caps in the header, the headuc attribute may be set to use the all caps \GLS⟨field⟩ or \GLSxtr⟨field⟩ instead (such as \GLStext or \GLSxtrshort).

There is currently only support for the name, text, plural, first, firstplural, short, shortplural, long, and longplural fields, and also limited support for the full form of abbreviations. For other fields, you will need to follow the recommendation of the glossaries manual (as discussed above in §5.3.1).

The previous example can be rewritten as follows:

\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\glsdefpostlink{general}{\glsxtrpostlinkAddDescOnFirstUse}
\newglossaryentry{sample}{name={sample},description={an example}}
\newglossaryentry{alpha}{name={\ensuremath{\alpha}},description={alpha}}
\begin{document}
\tableofcontents
\section{\Glsfmttext{sample} and \glsfmttext{alpha}}
First use: \gls{sample} and \gls{alpha}.
\end{document}
5. Referencing (Using) Entries

Next use: \gls{sample} and \gls{alpha}.

\printunsrtglossary
\end{document}

Example 93: References in section headings using \glsfmttext

Contents

1  Sample and $\alpha$  1

Glossary

1  Sample and $\alpha$

First use: sample (an example) and $\alpha$ (alpha).

Next use: sample and $\alpha$.

Glossary

sample an example

$\alpha$ alpha

Note that this still results in “Token not allowed in a PDF string” warnings from hyperref. This is due to the maths shift and $\alpha$, and is something that would also occur if the section title explicitly contained $\alpha$. If this is likely to happen, the issue can be solved by placing \texorpdfstring within the field value. For example:

\glsnoexpandfields
\newglossaryentry{alpha}{description={alpha}, name={\texorpdfstring{\ensuremath{\alpha}}{alpha}}}

Note the need to prevent field expansion with \glsnoexpandfields, otherwise \texorpdfstring will be prematurely expanded while the entry is being defined.

The options noindex and hyper=false are hard-coded when the commands listed below, such as \glsfmtshort, occur in the header or contents, but within the actual section title or
caption in the document text, those options are obtained from:

\glsxtrtitleopts

This simply expands to the option list. For example, you may actually want a hyperlink and indexing to occur in the document body, in which case redefine \glsxtrtitleopts to do nothing:

\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\pagestyle{headings}
\glssetcategoryattribute{general}{headuc}{true}
\glsdefpostlink{general}{\glsxtrpostlinkAddDescOnFirstUse}
\newglossaryentry{sample}{name={sample},description={an example}}
\renewcommand{\glsxtrtitleopts}{ }
\begin{document}
\section{\Glsfmttext{sample}}
First use: \gls{sample}. Next use: \gls{sample}.
\printunsrtglossary
\end{document}

Example 94: Reference with hyperlink in section headings

1  SAMPLE

1  Sample
First use: sample (an example). Next use: sample.

Glossary

sample  an example
5. Referencing (Using) Entries

\glsfmtshort\{\entry-label\}

This normally behaves like \glsxtrshort but expands to just \glsentryshort in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtshort\{\entry-label\}

This normally behaves like \Glsxtrshort but expands to:

\MFUsentencecase{\glsentryshort\{\entry-label\}}

in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtshort\{\entry-label\}

This normally behaves like \GLSxtrshort but expands to just \glsentryshort in PDF bookmarks and is adjusted when appearing in the header or contents.

\pglsfmtshort[\options]\{\entry-label\}[\{insert\}] modifiers: * + (alt-mod)

As \glsfmtshort but inserts the prefix field and separator, if the prefix value is set and non-empty. Provided for use with glossaries-prefix.

\Pglssfmtshort[\options]\{\entry-label\}[\{insert\}] modifiers: * + (alt-mod)

As \pglsfmtshort but sentence case. Note the initial “P” in the command name, which matches \Pgls (similarly for the other prefix sentence case commands).

\PGLSfmtshort[\options]\{\entry-label\}[\{insert\}] modifiers: * + (alt-mod)

As \pglsfmtshort but all caps.

\glsfmtshortpl\{\entry-label\}

This normally behaves like \glsxtrshortpl but expands to just \glsentryshortpl in PDF bookmarks and is adjusted when appearing in the header or contents.
bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtshortpl{⟨entry-label⟩}

This normally behaves like \Glsxtrshortpl but expands to:

\MFUsentencecase{\glsentryshortpl{⟨entry-label⟩}}

in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtshortpl{⟨entry-label⟩}

This normally behaves like \GLXstrshortpl but expands to just \glsentryshortpl in PDF bookmarks and is adjusted when appearing in the header or contents.

\pglsfmtshortpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \glsfmtshortpl but inserts the prefixplural field and separator, if the prefixplural value is set and non-empty. Provided for use with glossaries-prefix.

\Pglssfmtshortpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \pglsfmtshortpl but sentence case.

\PGLSfmtshortpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

As \pglsfmtshortpl but all caps.

\glsfmtlong{⟨entry-label⟩}

This normally behaves like \glsxtrlong but expands to just \glsentrylong in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glssfmtlong{⟨entry-label⟩}

This normally behaves like \Glsxtrlong but expands to:
5. Referencing (Using) Entries

\MFU{}, {\glsentrylong{(entry-label)}}

in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtlong{(entry-label)}

This normally behaves like \GLSxtrlong but expands to just \glsentrylong in PDF bookmarks and is adjusted when appearing in the header or contents.

\pgrlsfmtlong[(options)]{(entry-label)}[(insert)] modifiers: * + ⟨alt-mod⟩

As \glsfmtlong but inserts the \texttt{prefixfirst} field and separator, if the \texttt{prefixfirst} value is set and non-empty. Provided for use with glossaries–prefix.

\PGLSfmtlong[(options)]{(entry-label)}[(insert)] modifiers: * + ⟨alt-mod⟩

As \pgrlsfmtlong but sentence case.

\PGLSfmtlongpl[(options)]{(entry-label)}[(insert)] modifiers: * + ⟨alt-mod⟩

As \pgrlsfmtlong but all caps.

\glsfmtlongpl{(entry-label)}

This normally behaves like \GLSxtrlongpl but expands to just \glsentrylongpl in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtlongpl{(entry-label)}

This normally behaves like \GLSxtrlongpl but expands to:

\MFU{}, {\glsentrylongpl{(entry-label)}}

in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtlongpl{(entry-label)}
This normally behaves like \GLSxtrlongpl but expands to just \glsentrylongpl in PDF bookmarks and is adjusted when appearing in the header or contents.

\pglsfmtlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩}  modifiers: * + ⟨alt-mod⟩

As \pglsfmtlongpl but inserts the prefixfirstplural field and separator, if the prefix-firstplural value is set and non-empty. Provided for use with glossaries-prefix.

\Pglsfmtlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩}  modifiers: * + ⟨alt-mod⟩

As \pglsfmtlongpl but sentence case.

\PGLSfmtlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩}  modifiers: * + ⟨alt-mod⟩

As \pglsfmtlongpl but all caps.

The full form is slightly different as it doesn’t correspond to an individual field but instead is formed from a combination of the short and long fields (the order depending on the abbreviation style). Since it’s too complicated to simply expand to the appropriate style, a simple expandable command is provided for the PDF bookmarks:

\glspdfformfull{⟨entry-label⟩}

This just expands to the long form followed by the short form in parentheses:

\newcommand*{\glspdfformfull}[1]{\glsentrylong{#1}(\glsentryshort{#1})}

You will need to redefine this if you require the short form first. There is an analogous command for the plural:

\glspdfformfullpl{⟨entry-label⟩}

This has a similar definition to \glspdfformfull but uses \glsentrylongpl and \glsentryshortpl.
This normally behaves like \glsxtrfull but expands to just \glspdffmtfull in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtfull{⟨entry-label⟩}

This normally behaves like \Glsxtrfull but expands to: \MFUsentencecase{\glspdffmtfull{⟨entry-label⟩}}
in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtfull{⟨entry-label⟩}

This normally behaves like \GLSxtrfull but expands to just \glspdffmtfull in PDF bookmarks and is adjusted when appearing in the header or contents.

\glsfmtfullpl{⟨entry-label⟩}

This normally behaves like \glsxtrfullpl but expands to just \glspdffmtfullpl in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtfullpl{⟨entry-label⟩}

This normally behaves like \Glsxtrfullpl but expands to: \MFUsentencecase{\glspdffmtfullpl{⟨entry-label⟩}}
in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtfullpl{⟨entry-label⟩}

This normally behaves like \GLSxtrfullpl but expands to just \glspdffmtfullpl in PDF bookmarks and is adjusted when appearing in the header or contents.

\glsfmtname{⟨entry-label⟩}
5. Referencing (Using) Entries

This normally behaves like \glsname but expands to just \glsentryname in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtname{(entry-label)}

This normally behaves like \Glsname but expands to:

\MFUsentencecase{\glsentryname{(entry-label)}}
in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtname{(entry-label)}

This normally behaves like \GLSname but expands to just \glsentryname in PDF bookmarks and is adjusted when appearing in the header or contents.

\glsfmttext{(entry-label)}

This normally behaves like \glstext but expands to just \glsentrytext in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmttext{(entry-label)}

This normally behaves like \Glstext but expands to:

\MFUsentencecase{\glsentrytext{(entry-label)}}
in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmttext{(entry-label)}

This normally behaves like \GLStext but expands to just \glsentrytext in PDF bookmarks and is adjusted when appearing in the header or contents.

\glsfmtplural{(entry-label)}
This normally behaves like \glsplural but expands to just \glsentryplural in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtplural{⟨entry-label⟩}

This normally behaves like \GLSplural but expands to just \glsentryplural in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtplural{⟨entry-label⟩}

This normally behaves like \glsfirst but expands to just \glsentryfirst in PDF bookmarks and is adjusted when appearing in the header or contents.

\glsfmtfirst{⟨entry-label⟩}

This normally behaves like \Glsfirst but expands to just \glsentryfirst in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtfirst{⟨entry-label⟩}

This normally behaves like \glsfmtfirstpl but expands to:

\MFUsentencecase{\glsentryfirstpl{⟨entry-label⟩}}

in PDF bookmarks and is adjusted when appearing in the header or contents.

\glsfmtfirstpl{⟨entry-label⟩}
This normally behaves like \glsfirstplural but expands to just \glsentryfirstplural in PDF bookmarks and is adjusted when appearing in the header or contents.

\Glsfmtfirstpl{\langle entry-label \rangle}

This normally behaves like \Glsfirstplural but expands to:

\MFUsentencecase{\glsentryfirstplural{\langle entry-label \rangle}}

in PDF bookmarks and is adjusted when appearing in the header or contents.

\GLSfmtfirstpl{\langle entry-label \rangle}

This normally behaves like \GLSfirstplural but expands to just \glsentryfirstplural in PDF bookmarks and is adjusted when appearing in the header or contents.

5.3.3. Advanced Commands

This section is intended for advanced users and package developers.

The commands described here are irrelevant if you use \glsxtrRevertMarks to restore the definitions of \markright, \markboth and \@starttoc. If you use \glsxtrRevertTocMarks, then this section is only applicable to \markright and \markboth.

If you need to know whether or not some code is inside a header or contents list, you can use:

\glsxtrifinmark{\langle true \rangle}{\langle false \rangle}

This does \langle true \rangle if the command occurs within \markright, \markboth or \@starttoc otherwise does \langle false \rangle.

If you need to know whether or not some code is inside a contents list (but not the header), you can use:

\glsxtrifintoc{\langle true \rangle}{\langle false \rangle}

This does \langle true \rangle if the command occurs within \@starttoc otherwise it does \langle false \rangle. (The modified definition of \@starttoc sets \glsxtrifintoc to \@firstoftwo at the start and to \@secondoftwo at the end.)
5. Referencing (Using) Entries

If you need to know whether or not some code is in the PDF bookmarks or heading, you can use:

\glsxtrtitleorpdforheading{⟨title⟩}{⟨PDF bookmarks⟩}{⟨heading⟩}

This does the applicable argument depending on whether the command occurs within a title/ caption or PDF bookmark or heading.

If this command occurs within the toc file, it will do its ⟨heading⟩ argument but if \glsxtrtitleorpdforheading expands while it’s being written to the toc file, then it will expand to ⟨title⟩. This can be illustrated in the following document:

\documentclass{report}
\usepackage{lipsum}
\usepackage{hyperref}
\usepackage{glossaries-extra}
\pagestyle{headings}
\begin{document}
\tableofcontents
\chapter{\glsxtrtitleorpdforheading{Title}{PDF}{Heading}\glsxtrifinmark{in mark}{not in mark}}
\lipsum
\chapter{\protect\glsxtrtitleorpdforheading{Title}{PDF}{Heading}\protect\glsxtrifinmark{in mark}{not in mark}}
\lipsum
\end{document}

In the first case, \glsxtrtitleorpdforheading expands as it’s being written to the toc file, so it expands to “Title”. In the second case, \glsxtrtitleorpdforheading is protected so that command is written to the toc file. On the next \LaTeX, when the table of contents is displayed, this command will expand to “Heading”, because it’s in the toc file. Similarly, in the first case, \glsxtrifinmark will expand to “not in mark” as it’s written to the toc file, but in the second case it’s expansion is prevented, so it will expand to “in mark” in the table of contents.

If gettitlestring has been loaded (used by nameref to provide \nameref) then adjustments for both \glsxtrtitleorpdforheading and \glsxtrifinmark will be added to \GetTitleStringDisableCommands, but bear in mind that you will need to use the following for it to have an effect:

\GetTitleStringSetup{expand}

The commands described in §5.3.2, such as \glsfmtshort, are essentially defined as:
5. Referencing (Using) Entries

\texttt{\\textbackslash texorpdfstring \{\glsxtrtitle\{field\}\{\langle entry-label\rangle\}\% title \{\glsentry\{field\}\{\langle entry-label\rangle\}\% bookmark}

If \texttt{\texorpdfstring} isn’t defined, then the definition is:

\texttt{\glsxtrtitle\{field\}\{\langle entry-label\rangle\}}

For example, \texttt{\glsfmtshort} is defined as (with hyperref):

\texttt{\newcommand*{\glsfmtshort}[1]{\texttt{\\textbackslash texorpdfstring{\glsxtrtitleshort{#1}}{\\glsentryshort{#1}}}}\%}

This ensures that \texttt{\glsfmtshort} expands to just \texttt{\glsentryshort} within the PDF bookmarks. Provided the field value doesn’t contain any problematic commands, this allows the actual value to be added to the bookmarks.

Unfortunately the case-changing commands can’t expand and therefore aren’t appropriate for the bookmarks (which need to be a PDF string). This means that the sentence case and all caps commands also use the unmodified field value for the bookmark. For example, \texttt{\Glsfmtshort} is defined as:

\texttt{\newcommand*{\Glsfmtshort}[1]{\\textbackslash texorpdfstring{\glsxtrtitleshort{#1}}{\\glsentryshort{#1}}}}\%

The \texttt{\glsxtrtitle\{field\}} set of commands all default to the corresponding \texttt{\glistext-}\ like command with the options given by \texttt{\glsxtrtitleopts} and an empty insert final argument. These title commands are redefined by \texttt{\glsxtrmarkhook} to the corresponding \texttt{\glsxtrhead\{field\}} command. These “head” commands use \texttt{\NoCaseChange} to prevent interference from page headers that convert to all caps (which can inappropriately convert the entry label to all caps). Instead, the \texttt{headuc} attribute needs to be set to \texttt{true} to use the appropriate all caps command. A shortcut command is provided to test for this attribute:

\texttt{\glsxtrifheaduc\{\langle entry-label\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}}

This is defined as:

\texttt{\newcommand*{\glsxtrifheaduc}[3]{\\glsxtrifintoc[\#3]{\\glsifattribute[\#1]{headuc}{true}{\#2}{\#3}}}}\%
Since the header commands also end up in the contents, where the all caps conversion should not be applied, the definition includes \glsxtrifintoc to skip the check in the contents.

\glsxtrtitleshort{(entry-label)}

The normal behaviour of \glsfmtshort. This is redefined by \glsxtrmarkhook to \glsxtrshort. The default is:

\glsxtrshort[noindex,hyper=false]{(entry-label)}[]

(This is performed indirectly via an internal command that ensures that \glsxtrtitleopts is expanded before being passed in the optional argument.)

\glsxtrheadshort{(entry-label)}

Used to display the short form in the page header. This is defined as:

\newcommand*{\glsxtrheadshort}[1]{% \protect\NoCaseChange {% \glsifattribute{#1}{headuc}{true}% {% \GLSxtrshort[noindex,hyper=false]{#1}[]}% {% \glsxtrshort[noindex,hyper=false]{#1}[]}% }%}

The sentence case commands also check the headuc attribute.

\glsxtrtitleshort{(entry-label)}

The normal behaviour of \Glsfmtshort. This is redefined by \glsxtrmarkhook to \Glsxtrheadshort. The default is:

\Glsxtrshort[noindex,hyper=false]{(entry-label)}[]

(Again, this is performed indirectly via an internal command that ensures that \glsxtr-
Referencing (Using) Entries

titleopts is expanded before being passed in the optional argument.)

\Glsxtrheadshort{⟨entry-label⟩}

Used to display the sentence case short form in the page header. This is defined as:

\newcommand*{\Glsxtrheadshort}[1]{% 
\protect\NoCaseChange
{\% 
  \glsifattribute{#1}{headuc}{true}\% 
  {\% 
    \GLSxtrshort[noindex,hyper=false]{#1}[]\% 
  }\% 
  {\% 
    \Glsxtrshort[noindex,hyper=false]{#1}[]\% 
  }\% 
}}%
}

\GLSxtrtitleshort{⟨entry-label⟩}

The normal behaviour of \GLSfmtshort. This is redefined by \glsxtrmarkhook to \GLSxtrheadshort. The default uses \GLSxtrshort in a similar way to \glsxtrtitleshort and \Glsxtrtitleshort.

\GLSxtrheadshort{⟨entry-label⟩}

Used to display the all caps short form in the page header. In this case, there’s no need to check to the headuc attribute, but the label needs to be protected from any potential case-change:

\newcommand*{\GLSxtrheadshort}[1]{% 
\protect\NoCaseChange
{\% 
  \GLSxtrshort[noindex,hyper=false]{#1}[]\% 
}}%
}

All the similar commands listed below are defined in an analogous way, except for the glossaries–prefix commands, where only the sentence case title version is provided. This is
5. Referencing (Using) Entries

because commands like \Pglsfmtshort have to determine whether or not to use \glsfmtshort or \Glsfmtshort depending on whether or not the prefix has been set. Whereas commands like \pglsfmtshort simply need to insert the prefix and separator if set and then use the corresponding \glsfmtshort.

\Pglsxtrtitleshort\{⟨entry-label⟩\}
The normal behaviour of \Pglsfmtshort.

\Pglsxtrtitleshorth\{⟨entry-label⟩\}
The normal behaviour of \Pglsfmtshorth.

\Pglsxtrtitle\{⟨entry-label⟩\}
The normal behaviour of \Pglsfmtlong.

\Pglsxtrtitlepl\{⟨entry-label⟩\}
The normal behaviour of \Pglsfmtlongpl.

\glsxtrtitleshorth\{⟨entry-label⟩\}
The title plural short form. (Normal behaviour of \glsfmtshorth.)

\glsxtrheadshorth\{⟨entry-label⟩\}
The header plural short form. (The behaviour of \glsfmtshorth when it occurs in a header.)

\Glsxtrtitleshorth\{⟨entry-label⟩\}
The title plural sentence case short form. (Normal behaviour of \Glsfmtshorth.)

\Glsxtrheadshorth\{⟨entry-label⟩\}
5. Referencing (Using) Entries

The header plural sentence case short form. (The behaviour of \Glsfmtshortpl when it occurs in a header.)

\GLSxtrtitleshortpl\{entry-label\}

The title plural all caps short form. (Normal behaviour of \Glsfmtshortpl.)

\GLSxtrheadshortpl\{entry-label\}

The header plural all caps short form. (The behaviour of \Glsfmtshortpl when it occurs in a header.)

\glsxtrtitlelong\{entry-label\}

The title long form. (Normal behaviour of \glsxtrlong.)

\glsxtrheadlong\{entry-label\}

The header long form. (The behaviour of \glsxtrlong when it occurs in a header.)

\Glsxtrtitlelong\{entry-label\}

The title sentence case long form. (Normal behaviour of \Glsfmtlong.)

\Glsxtrheadlong\{entry-label\}

The header sentence case long form. (The behaviour of \Glsfmtlong when it occurs in a header.)

\GLSxtrtitlelong\{entry-label\}

The title all caps long form. (Normal behaviour of \Glsfmtlong.)

\GLSxtrheadlong\{entry-label\}
5. Referencing (Using) Entries

The header all caps long form. (The behaviour of \GLSfmtlong when it occurs in a header.)

\glsxtrtitlelongpl{⟨entry-label⟩}

The title plural long form. (Normal behaviour of \glsfmtlongpl.)

\glsxtrheadlongpl{⟨entry-label⟩}

The header plural long form. (The behaviour of \glsfmtlongpl when it occurs in a header.)

\Glsxtrtitlelongpl{⟨entry-label⟩}

The title plural sentence case long form. (Normal behaviour of \Glstitlelongpl.)

\Glsxtrheadlongpl{⟨entry-label⟩}

The header plural sentence case long form. (The behaviour of \Glstitlelongpl when it occurs in a header.)

\GLSxtrtitlelongpl{⟨entry-label⟩}

The title plural all caps long form. (Normal behaviour of \GLSfmtlongpl.)

\GLSxtrheadlongpl{⟨entry-label⟩}

The header plural all caps long form. (The behaviour of \GLSfmtlongpl when it occurs in a header.)

\glsxtrtitlefull{⟨entry-label⟩}

The title full form. (Normal behaviour of \glsfmtfull.)

\glsxtrheadfull{⟨entry-label⟩}
5. Referencing (Using) Entries

The header full form. (The behaviour of \glsfmtfull when it occurs in a header.)

\Glxtrtitlefull{⟨entry-label⟩}

The title sentence case full form. (Normal behaviour of \Glsfmtfull.)

\Glsxtrheadfull{⟨entry-label⟩}

The header sentence case full form. (The behaviour of \Glsfmtfull when it occurs in a header.)

\GLSxtrtitlefull{⟨entry-label⟩}

The title all caps full form. (Normal behaviour of \GLSfmtfull.)

\GLSxtrheadfull{⟨entry-label⟩}

The header all caps full form. (The behaviour of \GLSfmtfull when it occurs in a header.)

\glsxtrtitlefullpl{⟨entry-label⟩}

The title plural full form. (Normal behaviour of \glsfmtfullpl.)

\glsxtrheadfullpl{⟨entry-label⟩}

The header plural full form. (The behaviour of \glsfmtfullpl when it occurs in a header.)

\Glsxtrtitlefullpl{⟨entry-label⟩}

The title plural sentence case full form. (Normal behaviour of \Glsfmtfullpl.)

\Glsxtrheadfullpl{⟨entry-label⟩}

The header plural sentence case full form. (The behaviour of \Glsfmtfullpl when it occurs
5. Referencing (Using) Entries

in a header.)

\GLSxtrtitlefullpl{(entry-label)}
The title plural all caps full form. (Normal behaviour of \GLSfmtfullpl.)

\GLSxtrheadfullpl{(entry-label)}
The header plural all caps full form. (The behaviour of \GLSfmtfullpl when it occurs in a header.)

\glsxtrtitlename{(entry-label)}
The title \textit{name} field. (Normal behaviour of \glssfmtname.)

\glsxtrheadname{(entry-label)}
The header \textit{name} field. (The behaviour of \glssfmtname when it occurs in a header.)

\Glssxtrtitlename{(entry-label)}
The title sentence case \textit{name} field. (Normal behaviour of \Glssfmtname.)

\Glssxtrheadname{(entry-label)}
The header sentence case \textit{name} field. (The behaviour of \Glssfmtname when it occurs in a header.)

\GLSxtrtitlefullpl{(entry-label)}
The title all caps \textit{name} field. (Normal behaviour of \GLSfmtname.)

\GLSxtrheadfullpl{(entry-label)}
5. Referencing (Using) Entries

The header all caps name field. (The behaviour of \GLSfmtname when it occurs in a header.)
\glsxtrtitletext{⟨entry-label⟩}

The title text field. (Normal behaviour of \glsfmttext.)
\glsxtrheadtext{⟨entry-label⟩}

The header text field. (The behaviour of \glsfmttext when it occurs in a header.)
\GLSxtrtitletext{⟨entry-label⟩}

The title sentence case text field. (Normal behaviour of \GLSfmttext.)
\GLSxtrheadtext{⟨entry-label⟩}

The header sentence case text field. (The behaviour of \GLSfmttext when it occurs in a header.)
\GLSxtrtitletext{⟨entry-label⟩}

The title all caps text field. (Normal behaviour of \GLSfmttext.)
\GLSxtrheadtext{⟨entry-label⟩}

The header all caps text field. (The behaviour of \GLSfmttext when it occurs in a header.)
\glsxtrtitleplural{⟨entry-label⟩}

The title plural field. (Normal behaviour of \glsfmtplural.)
\glsxtrheadplural{⟨entry-label⟩}
5. Referencing (Using) Entries

The header **plural** field. (The behaviour of `\glsfmtplural` when it occurs in a header.)

\Glsxtrtitleplural\{⟨entry-label⟩\}

The title sentence case **plural** field. (Normal behaviour of `\Glsfmtplural`.)

\Glsxtrheadplural\{⟨entry-label⟩\}

The header sentence case **plural** field. (The behaviour of `\Glsfmtplural` when it occurs in a header.)

\GLSxtrtitleplural\{⟨entry-label⟩\}

The title all caps **plural** field. (Normal behaviour of `\GLSfmtplural`.)

\GLSxtrheadplural\{⟨entry-label⟩\}

The header all caps **plural** field. (The behaviour of `\GLSfmtplural` when it occurs in a header.)

\glsxtrtitlefirst\{⟨entry-label⟩\}

The title **first** field. (Normal behaviour of `\glsfmtfirst`.)

\glsxtrheadfirst\{⟨entry-label⟩\}

The header **first** field. (The behaviour of `\glsfmtfirst` when it occurs in a header.)

\Glsxtrtitlefirst\{⟨entry-label⟩\}

The title sentence case **first** field. (Normal behaviour of `\Glsfmtfirst`.)

\Glsxtrheadfirst\{⟨entry-label⟩\}

The header sentence case **first** field. (The behaviour of `\Glsfmtfirst` when it occurs in a header.)
5. Referencing (Using) Entries

\Glsxtrtitlefirst\{(entry-label)\}

The title all caps \textit{first} field. (Normal behaviour of \GLSfmtfirst.)

\Glsxtrheadfirst\{(entry-label)\}

The header all caps \textit{first} field. (The behaviour of \GLSfmtfirst when it occurs in a header.)

\glsxtrtitlefirstplural\{(entry-label)\}

The title \textit{firstplural} field. (Normal behaviour of \GLSfmtfirstpl.)

\glsxtrheadfirstplural\{(entry-label)\}

The header \textit{firstplural} field. (The behaviour of \GLSfmtfirstpl when it occurs in a header.)

\Glsxtrtitlefirstplural\{(entry-label)\}

The title \textit{sentence case firstplural} field. (Normal behaviour of \Glsfmtfirstpl.)

\Glsxtrheadfirstplural\{(entry-label)\}

The header \textit{sentence case firstplural} field. (The behaviour of \Glsfmtfirstpl when it occurs in a header.)

\GLSxtrtitlefirstplural\{(entry-label)\}

The title all caps \textit{firstplural} field. (Normal behaviour of \GLSfmtfirstpl.)

\GLSxtrheadfirstplural\{(entry-label)\}

The header all caps \textit{firstplural} field. (The behaviour of \GLSfmtfirstpl when it occurs in a header.)

226
5. Referencing (Using) Entries

The definitions of \markright, \markboth and \@starttoc are saved (using \let) when glossaries-extra loads.

\@glsxtr@org@markright{⟨text⟩}
The previous definition of \markright.

\@glsxtr@org@markboth{⟨left text⟩}{⟨right text⟩}
The previous definition of \markboth.

\@glsxtr@org@@starttoc{⟨toc⟩}
The previous definition of \@starttoc.

The glossaries-extra definitions of \markright, \markboth and \@starttoc all start and end with hooks that redefine commands that are sensitive to being in the header or contents.

\glsxtrmarkhook
This saves the original definitions and redefines the sensitive commands. This includes \MakeUppercase which is \let to \MakeTextUppercase.

\@glsxtrinmark
This redefines \glsxtrifinmark to just do its first argument (⟨true⟩).

\@glsxtrnotinmark
This redefines \glsxtrifinmark to just do its second argument (⟨false⟩).

\glsxtrrestoremarkhook
This restores the sensitive commands to the saved definitions. (For use where grouping will cause interference.) For example, \markboth is redefined as:
5.4. Nested Links

Complications arise when you use the `\gls`-like commands in the value of the `name` field (or `text` or `first` fields, if set). This tends to occur with abbreviations that extend other abbreviations. For example, SHTML is an abbreviation for SSI enabled HTML, where SSI is an abbreviation for Server Side Includes and HTML is an abbreviation for Hypertext Markup Language.

For example, things can go wrong if the following is used with the `glossaries` package:

\begin{verbatim}
\newacronym{ssi}{SSI}{Server Side Includes}
\newacronym{html}{HTML}{Hypertext Markup Language}
\newacronym{shtml}{S\gls{html}}{\gls{ssi} enabled \gls{html}}
\end{verbatim}

The main problems are:

1. With older versions of `mfirstuc` and `glossaries`, the sentence case commands, such as `\Gls`, won’t work for the `shtml` entry on first use if the long form is displayed before the short form (which is the default abbreviation style). This will attempt to do

\begin{verbatim}
\gls{\uppercase ssi} enabled \gls{html}
\end{verbatim}

which just doesn’t work. Grouping the `\gls{ssi}` doesn’t work either as this will effectively try to do:

\begin{verbatim}
\uppercase{\gls{ssi}} enabled \gls{html}
\end{verbatim}

This will upper case the label `ssi` so the entry won’t be recognised. This problem will also occur if you use the all caps version, such as `\GLS{shtml}`.

With `mfirstuc` v2.08+ and `glossaries` v1.49+, this issue should now be resolved for sentence case where `\gls{ssi}` will be mapped to `\Gls{ssi}` within `\Gls{shtml}`. The all caps command `\GLS{shtml}` will treat `\gls` as an exclusion and so won’t perform a case-change. See §5.2 for further details.
5. Referencing (Using) Entries

2. The long and abbreviated forms accessed through \glsentrylong and \glsentryshort are no longer expandable and so can’t be used in contexts that require this, such as PDF bookmarks.

3. The nested commands may end up in the sort key, which will confuse the indexing.

4. The \shtml entry produces inconsistent results depending on whether the ssi or html entries have been used. Suppose both ssi and html are used before shtml. For example:

   - This section discusses \gls{ssi}, \gls{html} and \gls{shtml}.
   - This section discusses server side includes (SSI), hypertext markup language (HTML) and SSI enabled HTML (SHTML).

   In the above, the first use of the shtml entry produces “SSI enabled HTML (SHTML)”. Now let’s suppose the html entry is used before the shtml but the ssi entry is used after the shtml entry, for example:

   - The sample files are either \gls{html} or \gls{shtml}, but let’s first discuss \gls{ssi}.
   - The sample files are either hypertext markup language (HTML) or server side includes (SSI) enabled HTML (SHTML), but let’s first discuss SSI.

   In this case, the first use of the shtml entry now produces “server side includes (SSI) enabled HTML (SHTML)”, which looks a bit cumbersome.

   Now let’s suppose the shtml entry is used before (or without) the other two entries:

   - This article is an introduction to \gls{shtml}.
   - This article is an introduction to server side includes (SSI) enabled hypertext markup language (HTML) (SHTML).

   Now the first use of the shtml entry produces “server side includes (SSI) enabled hypertext markup language (HTML) (SHTML)”, which looks strange.

   This is all aggravated when using just the base glossaries package when the acronym style is set with \setacronymstyle. For example:

   \setacronymstyle{long-short}
as this references the label through the use of \glslabel when displaying the long and short forms, but this value changes with each use of \gls, so instead of displaying “(SHTML)” at the end of the first use, it now displays “(HTML)”, since \glslabel has been changed to html by \gls{html}.

Another oddity occurs if you reset the html entry between uses of the shtml entry. For example:

\gls{html} ... \glsreset{html}\gls{shtml}

The next use of shtml produces “Shypertext markup language (HTML)”, which is downright weird. (This is a result of the short form being set to S\gls{html}, but \gls{html} is showing the full form.)

Even without this, the short form has nested formatting commands, which amount to \textacronymfont{S\textacronymfont{HTML}}. This may not be a problem for some styles, but if you use one of the “sm” styles (that use \textsmaller), this will produce an odd result.

5. Each time the shtml entry is used, the html entry will also be indexed and marked as used, and on first use this will happen to both the ssi and html entries. This kind of duplication in the location list isn’t usually particularly helpful to the reader.

6. If hyperref is in use, you’ll get nested hyperlinks and there’s no consistent way of dealing with this across the available PDF viewers. If on the first use case, the user clicks on the “HTML” part of the “SSI enabled HTML (SHTML)” link, they may be directed to the HTML entry in the glossary or they may be directed to the SHTML entry in the glossary.

For these reasons, with just the base glossaries package, it’s better to use the simple expandable commands like \glsentrytext or \glsentryshort in the definition of other entries. The glossaries-extra package provides two other ways of dealing with these problems:

1. If the term can simply be treated as a series of previously defined entries, then consider using multi-entries (or compound sets), as described in §7. This deals with all the issues, including case-changing.

2. Use the partially-expandable \glsxtrp, described below.
where \( \langle \text{field} \rangle \) is the internal field label.

This command partially expands, so it will expand to just the field value if it occurs in the PDF bookmarks. Otherwise it will behave much like the commands described in §5.3.2, but with additional outer scoping and the post-link hook is suppressed.

Rather than testing the existence of the given field, this tests the existence of \texttt{\gls{field}} or \texttt{\glsxtr{field}}, which means that it may be confused if the \( \langle \text{field} \rangle \) argument is set to something that isn’t a field but happens to match either of those command names (such as \texttt{full}).

Note that, as with commands like \texttt{\glsfmtshort}, there’s no optional argument. The default settings are noindex and hyper=false. You can change this with:

\[
\glsxtrsetpopts{\langle \text{options} \rangle}
\]

The argument should be the new default options.

At the start of each glossary, the default options are locally changed with:

\[
\glossxtrsetpopts
\]

This is defined as:

\[
\newcommand*{\glossxtrsetpopts}{% \\
\glsxtrsetpopts{noindex}% 
}\]

This allows hyperlinks for any instance of \texttt{\glsxtrp} that occurs in the name or description, where it shouldn’t be problematic.

There are some shortcut commands for the most common fields:

\[
\glsps{\langle \text{entry-label} \rangle}
\]

which is equivalent to \texttt{\glsxtrp{short}{\langle \text{entry-label} \rangle}}, and

\[
\glspt{\langle \text{entry-label} \rangle}
\]

which is equivalent to \texttt{\glsxtrp{text}{\langle \text{entry-label} \rangle}}.
5. Referencing (Using) Entries

There are also sentence case and all caps versions.

\Glsxtrp{\langle field\rangle}{\langle entry-label\rangle}

This uses the corresponding sentence case command \Gls{field} or \Glsxtr{field}.

\GLSxtrp{\langle field\rangle}{\langle entry-label\rangle}

This uses the corresponding all caps command \GLS{field} or \GLSxtr{field}.

For example:

\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\setabbreviationstyle{long-em-short-em}
\newabbreviation{html}{HTML}{hypertext markup language}
\newabbreviation{ssi}{SSI}{server-side includes}
\newabbreviation{shtml}{SHTML}{\glsps{html} enabled}
\glsps{ssi}
\begin{document}
\tableofcontents
\section{\glsfmtlong{shtml}}
First use: \gls{shtml}, \gls{html}, \gls{ssi}.

Next use: \gls{shtml}, \gls{html}, \gls{ssi}.

\printunsrtglossaries
\end{document}
5. Referencing (Using) Entries

Example 95: Nested link text with `\glspl`

**Contents**

1 HTML *enabled* SSI 1

**Glossary**

1 HTML *enabled* SSI

First use: HTML *enabled* SSI (*SHTML*), *hypertext markup language* (*HTML*), *server-side includes* (*SSI*).

Next use: *SHTML*, *HTML*, *SSI*.

**Glossary**

*HTML* hypertext markup language

*SSI* server-side includes

*SHTML* HTML *enabled* SSI

The way that this works is as follows:

- `\glsfmtlong{shtml}` expands to `\glsentrylong{shtml}` within the PDF bookmarks, which expands to the value of the *long* field:

```
\glsps{html} enabled \glsps{ssi}
```

This means that `\glsps` (within the PDF bookmarks) in turn expands to `\glsentryshort`. So the bookmark text (which can’t contain any formatting commands) ends up as "HTML enabled SSI".

- `\glsfmtlong{shtml}` essentially behaves like `\glsxtrlong`, but with the indexing and hyperlink suppressed. The link text is the value of the *long* field encapsulated with the abbreviation style’s formatting command (`\glslongemfont` in this case):

```
\glslongemfont{\glsps{html} enabled \glsps{ssi}}
```

This then becomes:
5. Referencing (Using) Entries

Note the grouping and localised suppression of the post-link hook.

Note that in the above example, with older versions of mfirstuc and glossaries, it’s not possible to use \Glsxtrlong{shtml} or similar. The problem here is that it will attempt to do:

```
\makefirstuc{\glsps{html} enabled \glsps{ssi}}
```

This will essentially end up as:

```
\glsps{\uppercase html} enabled \glsps{ssi}
```

which doesn’t work. If you want to protect against automated case-changes, such as using the glossdesc attribute, insert an empty brace at the start:

```
\newabbreviation{shtml}{SHTML}{{}\glsps{html} enabled \glsps{ssi}}
```

Alternatively, upgrade to mfirstuc v2.08+ and glossaries v4.50+. See §5.2.

5.5. Adjusting the Text Style

The \gls-like and \glstext-like commands produce text that’s essentially formatted either as (hyperoutside=true):

```
⟨hyper-cs⟩{⟨target⟩}{⟨textformat-cs⟩{⟨content⟩}}⟨post-link hook⟩
```

or (hyperoutside=false):

```
⟨textformat-cs⟩{⟨hyper-cs⟩{⟨target⟩}{⟨content⟩}}⟨post-link hook⟩
```

If hyperlinks are enabled then ⟨hyper-cs⟩ creates the hyperlink based on ⟨target⟩ with the hyperlink text given by the second argument. If hyperlinks aren’t enabled then ⟨hyper-cs⟩ ignores the ⟨target⟩ argument and simply does the second argument.

The ⟨content⟩ part is the link text, which includes the final optional ⟨insert⟩ (if supplied). The actual content depends on the command used (for example, \gls or \glstext). The \gls-like commands all use the entry display style associated with the entry’s glossary type, (see §5.5.5). The \glstext-like commands set the ⟨content⟩ to the corresponding field value.
with the insert appended, all encapsulated with the inner formatting (see §5.5.3), with appropriate case-changing, if required.

The abbreviation commands \glsxtrshort, \glsxtrlong, \glsxtrfull etc are considered part of the set of \glstext-like commands, but the content is set according to the abbreviation style (see §4.5).

The commands \glsdisp and \glslink both have the content part explicitly set in their final argument. There’s no insert optional argument as it can simply be included in the content part. The difference between them is that \glsdisp is considered a \gls-like command (it unsets the first use flag, §5.10, and uses the entry display style, §5.5.5), whereas \glslink is considered a \glstext-like command.

The \textit{⟨post-link hook⟩} part is described in §5.5.4.

The \textit{⟨textformat-cs⟩} command is the outer formatting command, described in §5.5.1. This doesn’t include the post-link hook. If you want to include the post-link hook then you need to encapsulate the entire \gls-like and \glstext-like command (including the final optional argument, if present, and following punctuation, if the post-link hook looks ahead for punctuation).

Some sensitive formatting commands need to have the actual text in their argument (or else have the argument in an unbreakable box). The \textit{⟨content⟩} part is usually too complicated for these commands. To help support this type of command, there is also an inner format, which is described in §5.5.3. In general, unless you require one of these sensitive commands, avoid setting the inner text format as it requires support from the underlying style (either the entry format style or the abbreviation style), which may not be available.

The following example document is ugly, but demonstrates the outer formatting (type-writer font), middle formatting (bold for regular entries and italic for abbreviations), inner formatting (highlighted in yellow), hyperlinks (red), and the category post-link hook (the description follows in parentheses for general entries on first use).
5. Referencing (Using) Entries

\% define entries:
\newglossaryentry{sample}{name={sample},description={an example}}
\newabbreviation{html}{HTML}{hypertext markup language}
\newacronym{nasa}{NASA}{National Aeronautics and Space Administration}
\begin{document}
First use: \gls{sample}, \gls{html}, \gls{nasa}.

Next use: \gls{sample}, \gls{html}, \gls{nasa}.
\end{document}

This produces:

Example 96: Link text styles: outer, middle, inner, hyperlinks and post-link hook
First use: \textit{sample (an example)}, \textit{hypertext markup language (HTML)}, \textit{NASA}.
Next use: \textit{sample}, \textit{HTML}, \textit{NASA}.

\textbf{Glossary}

\begin{description}
\item[sample] \textit{an example}
\item[HTML] \textit{hypertext markup language}
\item[NASA] \textit{National Aeronautics and Space Administration}
\end{description}

Note that the hyperlink, outer and middle formatting aren’t applied to the post-link hook.
The acronym category has the short-nolong abbreviation style, which sets the regular attribute to true.
This means that the NASA entry uses the regular middle format (\glsxtrregularfont) not the abbreviation middle format (\glsxtrabbreviationfont).

If you have a formatting command that needs to have its argument fully-expanded before being applied, you may be able to use:

\GlsXtrExpandedFmt{⟨cs⟩}{⟨content⟩}

This fully-expands ⟨content⟩ and does ⟨cs⟩{⟨expanded-content⟩}, where ⟨cs⟩ is a command that takes a single argument. For example, to use soul’s underlining command \ul:
Referencing (Using) Entries

(See Example 117 on page 290.) This isn’t guaranteed to work as the link text may contain fragile content.

The inner formatting can be unpredictable. For example, abbreviation styles are complicated and so the inner formatting command is included in some of the field values, such as the name, which is why the abbreviation name is highlighted in the glossary. In the above example, the inner formatting is included in the category post-link hook, but only because \glsxtrpostlinkAddDescOnFirstUse is designed to include it. If the category post-link hook was simply defined as:

\glsdefpostlink{general}{%
\glsxtrifwasfirstuse{ (\glsentrydesc{\glslabel})}{}%
}

then the inner formatting won’t be applied, since it’s not included in the hook. This is demonstrated in a slightly modified version of the above document (initial part of preamble that deals with loading packages and redefining formatting commands as before):

% post-link hook for 'general' category:
\glsdefpostlink{general}{%
\glsxtrifwasfirstuse{ (\glsentrydesc{\glslabel})}{}%
}

% this style sets the post-link hook for 'abbreviation' category:
\setabbreviationstyle{long-postshort-user}

% this style sets the post-link hook for 'acronym' category:
\setabbreviationstyle[acronym]{short-postfootnote}

% define entries:
\newglossaryentry{sample}{name={sample},description={an example}}
\newabbreviation{html}{HTML}{hypertext markup language}
\newacronym{nasa}{NASA}{National Aeronautics and Space Administration}
\begin{document}
First use: \gls{sample}, \gls{html}, \gls{nasa}.
Next use: \gls{sample}, \gls{html}, \gls{nasa}.
\printunsrtglossaries
\end{document}

This produces:
Example 97: Link text styles: outer, middle, inner, hyperlinks and post-link hooks (custom and abbreviation style)

First use: sample (an example), hypertext markup language
Next use: sample, HTML, NASA

Glossary

sample  an example
HTML  hypertext markup language
NASA  National Aeronautics and Space Administration

The “post” abbreviation styles put some content into the post-link hook and provide support for the inner formatting. The above example sets the abbreviation style to long-postshort-user. This sets up the post-link hook for the associated category (abbreviation, in this case) to show the parenthetical material. Be aware that this will override any previous definition of that hook. This style supports the inner formatting (so the parenthetical material is highlighted).

Similarly, the short-postfootnote style is applied to the acronym category, and sets the post-link hook for that category (which looks head for punctuation). The inner formatting is applied to the footnote text but not the marker.

The post-link hook for the general category is now much simpler and doesn’t include support for the inner formatting, so it’s not highlighted.

None of the post-link content is incorporated into the hyperlink, outer or middle formatting.

In general, it’s better to adjust the abbreviation’s style commands (see §4.5.1.3) rather than use the middle or inner formatting if abbreviations need to be displayed in a particular font.

5.5.1. Outer Formatting

By default, the outer formatting is produced with \glisttextformat, which is defined by the base glossaries package. However it can be replaced by the textformat category attribute or
by the `textformat` option. The order of precedence (not cumulative) is: the option supplied to the \gls-like or \glstext-like command, the category attribute, `\glstextformat`. For example:

\begin{verbatim}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\glsdefpostlink{general}{ (<\glsentrydesc{\glslabel})}
\newglossaryentry{sample}{name={sample},description={an example}}
\newcommand{\strong}{\textbf{\color{green}#1}}
\renewcommand{\glstextformat}{\emph{#1}}
\begin{document}
\gls{sample}\text{-insert}. \strong{\gls{sample}\text{-insert}}.
\glssetcategoryattribute{general}{textformat}{\textbf{}}
\gls{sample}\text{-insert}.
\gls[hyperoutside=false]{sample}\text{-insert}.
\gls[\textformat=textsf]{sample}\text{-insert}.
\end{document}
\end{verbatim}

Example 98: Changing the outer text format

sample-insert (an example). sample-insert (an example).
sample-insert (an example). sample-insert (an example).
sample-insert (an example).

The red text colour is from the hyperlink (red is the default with hyperref’s `colorlinks` option). The green from the custom `\strong` command is cancelled by the hyperlink colour change when the hyperlink is inside `\strong`.

After the `textformat` attribute is set, the `\glstextformat` command isn’t used, which is why the remaining lines don’t have any italic. The final line uses the `textformat` option, which overrides the `textformat` attribute, so neither `\glstextformat` nor the custom `\strong` are used.

Note that the only time that the post-link hook is included in the formatting is when the entire \gls command has been encapsulated.

5.5.2. Middle Formatting

The middle formatting comes between the outer formatting (§5.5.1 above) and the inner formatting (§5.5.3 below).

The middle formatting is implemented by the entry format style (§5.5.5) for the \gls-like commands or is initialised by `\glsxtrassignfieldfont` for the `\glstext`-like commands.
5. Referencing (Using) Entries

If you provide your own custom entry format style you will need to add support for the middle formatting, if required.

\glsxtrregularfont{⟨text⟩}

The command to use for regular entries. This is initialised to just do its argument.

\glsxtrabbreviationfont{⟨text⟩}

The command to use for abbreviations that considered non-regular entries.

The following document has a regular entry (sample), a regular abbreviation (radar, which uses short-nolong the default acronym style), and a non-regular abbreviation (HTML, which uses long-short the default abbreviation style):

\newglossaryentry{sample}{name={sample},description={an example}}
\newabbreviation{html}{HTML}{hypertext markup language}
\newacronym{radar}{radar}{radio detection and ranging}
\renewcommand{\glsxtrregularfont}{\emph{#1}}
\renewcommand{\glsxtrabbreviationfont}{\textbf{#1}}
\begin{document}
\gls{sample}, \gls{html}, \gls{radar}.
\end{document}

Example 99: Middle formatting

sample, hypertext markup language (HTML), radar.

Note that even though radar is an abbreviation, it’s considered a regular entry because it uses a regular style.

\glsxtrassignfieldfont{⟨entry-label⟩}

This command is used by all the \glstext-like commands to initialise the internal command used to encapsulate the field value. This will either be set to \glsxtrregularfont (for regular entries) or \@firstoftwo otherwise.

Note that this doesn’t use \glsxtrabbreviationfont as non-regular abbreviations are too complicated to work with \glstext, \glstfirst, \glstplural, \glstfirstplural or their case-changing variants. Instead, use the \glS-like commands or the abbreviation commands, such as \glsxtrshort.
5. Referencing (Using) Entries

5.5.3. Inner Formatting

If you want to format the link text, the best method is to either use the outer formatting or encapsulate the entire \gls-like or \glstext-like command, as described in §5.5.1. However, there are some sensitive commands that don’t work if the command argument doesn’t simply contain text.

Sometimes the issue may occur when the sensitive command that needs to encapsulate \gls doesn’t like boolean variables being changed (which occurs when the first use flag is unset). If this is the case, you may want to consider buffering as an alternative (see §5.10.1).

For example, if the sample document from §5.5.1 is adjusted to include the soul package and the following line is added to the document:

\gls[textformat=hl]{sample}

then the document build will fail with the error:

! Package soul Error: Reconstruction failed.

Once solution is to do the following instead:

\hl{\mbox{\gls{sample}}}

This will now work, but the box will prevent hyphenation, so it’s only useful if the link text is short, such as a symbol. If the link text is long (such as a phrase or the first use of an abbreviation), this method can produce undesirable results with overfull or underfull lines.

The inner formatting is designed to provide a workaround, but it must be implemented deep within the entry style formatting. This means that if you provide your own custom style, you will need to add the appropriate commands if you want that style to support inner formatting. You may also need to switch to using \MFUsentencecase instead of \makefirstuc if any of the sentence case commands are required:

\renewcommand{\glssentencecase}[1]{\MFUsentencecase{#1}}

Although there’s no guarantee that this will work for some particularly problematic formatting commands.

With the default entry style, the above example can be changed to:
5. Referencing (Using) Entries

\gls[innertextformat=hl]{sample}

The inner formatting may be split up in order to move them into the arguments of internal commands, such as those used for case-changing. This can result in unwanted side-effects.

The following uses \fbox (which draws a frame around its argument) and soul’s \so (which spaces out the letters):

```
% requires glossaries.sty v4.50+ and mfirstuc v2.08+
\renewcommand{\glssentencecase}{\MFUsentencecase{#1}}
\newacronym{radar}{radar}{radio detection and ranging}
\begin{document}
\Gls{radar}'s system\ldots
\Gls[innertextformat=fbox]{radar}'s system\ldots
\fbox{\Gls{radar}'s} system\ldots
\so{\mbox{\Gls{radar}'s}} system\ldots
\end{document}
```

Example 100: Inner formatting

```
\glslink{radar}{\so{Radar's}} system\ldots
```

Note the fragmentation of the inner formatting. The use of \mbox in the final line prevents an error but the letters aren’t spaced out. The only way to deal with this case is to use \glsdisp or \glslink with the text explicitly written:

```
\glslink{radar}{\so{Radar's}} system\ldots
```

242
5. Referencing (Using) Entries

The above example requires mfirstuc v2.08+.

Below are the commands used to support inner formatting.

\glsxtrgenentrytextfmt

This is the command that’s used to encapsulate any content that should have the inner formatting applied. It should not be redefined within the document as it’s initialised within the \gls-like and \glstext-like commands. It’s used within \glsgenentryfmt and included in the helper commands used by the predefined abbreviation styles.

Sometimes it may be necessary to include \glsxtrgenentrytextfmt within the actual field value to ensure that it’s as close as possible to the text. This is performed automatically when an entry is defined if the encapinnerfmt or encapnocaseinnerfmt attributes are set. Note that even in this case, fragmentation will occur with sentence case commands like $\textsc{Gls}$ or with the insert optional argument, as in the above example with $\fbox{m}$ and $\so{

\glsxtrdefaultentrytextfmt{$⟨text⟩$}

This is the default command that \glsxtrgenentrytextfmt will be $\let$ to within the \gls-like and \glstext-like commands before their options are processed. This simply does its argument but may be redefined. (See Example 117 on page 290.)

\glsxtrattrentrytextfmt{$⟨text⟩$}

This command applies formatting according to whether or not the innertextformat attribute is set. It isn’t used by default as it should rarely be needed and increases complexity. However, if you would like to provide support for the innertextformat attribute, you can redefine \glsxtrdefaultentrytextfmt to use \glsxtrattrentrytextfmt:

\renewcommand{\glsxtrdefaultentrytextfmt}{\glsxtrattrentrytextfmt}

This command expects the entry label to be stored in $\textsc{Gls}label$ (from which it obtains the category label).

The $\textsc{Gls}$-like commands use \glsxtrgenentrytextfmt within \glsgenentryfmt for regular entries or within the abbreviation style commands for non-regular abbreviations (see
5. Referencing (Using) Entries

§5.5.5).

The \glstext-like commands all essentially perform the following steps:

1. Initialise the middle formatting command \textlangle field-font-cs \textrangle used for encapsulating the field with \glsxtrassignfieldfont (see §5.5.2).

2. If \glsifapplyinnerfmtfield indicates that the field value should be encapsulated by \glsxtrgenentrytextfmt, then this essentially does (or appropriate case-change equivalent):

\[
\textlangle field-font-cs \textrangle \textlangle entry-label \textrangle \textlangle internal-field \textrangle
\]

otherwise it does:

\[
\textlangle field-font-cs \textrangle \textlangle entry-label \textrangle \textlangle insert \textrangle \textlangle internal-field \textrangle
\]

(See §9 for the “access” commands.)

For example, the link text for \glstext is:

\[
\glsifapplyinnerfmtfield\textlangle entry-label \textrangle \textlangle insert \textrangle \textlangle internal-field \textrangle
\]

The \glsaccessfmt \textlangle field \textrangle commands internally use \glsfmtfield to apply the inner formatting.

\[
\glsifapplyinnerfmtfield\textlangle entry-label \textrangle \textlangle internal-field \textrangle \textlangle true \textrangle \textlangle false \textrangle
\]

This determines whether or not the field identified by its internal field label for the given entry should have its value encapsulated by the inner formatting command. False indicates
that the field value already contains the inner formatting command.

\glsexclapplyinnerfmtfield{⟨entry-label⟩}{⟨internal-field⟩}

Locally adds the given field identified by its internal field label to the exclusion list for the given entry.

\glsfmtfield{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}{⟨internal-field⟩}

This command applies the formatting command ⟨cs⟩ (which takes one argument) to the entry’s field value identified by the given internal field label, including ⟨insert⟩ appended. This ensures that the internal control sequence used to store the field’s value is expanded before ⟨cs⟩ is applied.

\GLSfmtfield{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}{⟨internal-field⟩}

As above but sentence case.

\GLSfmtfield{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}{⟨internal-field⟩}

As above but all caps.

5.5.4. Post Link Hook

The post-link hook is a convenient way of automatically appending content after each instant of the \gls-like and \glstext-like commands. The simplest method of implementing this is with the category post-link hook, which is only applied to entries that have the given category. For example, the following will place an asterisk (*) after all entries with the default general category:

\glsdefpostlink{general}{*}
\newglossaryentry{sample}{name={sample}, symbol={X},
  description={an example}}
\begin{document}
\Gls{sample}, \glstext{sample}, \glsdesc{sample}
and \glssymbol{sample}.
\end{document}
5. Referencing (Using) Entries

Example 101: Category post-link hook

Sample*, sample*, an example* and X*.

Typically, the category post-link hook is more likely to include some conditional, such as to only insert text on first use. For example, `\glsxtrpostlinkAddDescOnFirstUse` can be used to insert the description in parentheses after the first use.

The “post” abbreviation styles all set the category post-link hook, which will overwrite any previous definition for the abbreviation’s category.

Within the post-link hook, you can use the placeholder commands, such as `\glslabel` (see §5.5.5), but note that you can’t use `\ifglsused` to determine whether or not the entry has been used, since the post-link hook comes after the entry has been unset. Instead, use `\glsxtrifwasfirstuse`. Additional commands provided for use within the post-link hooks are described in this section.

The post-link hook is implemented with `\glspostlinkhook`, which is defined by the base glossaries package. It’s used at the end of the `\gls`-like and `\glstext`-like commands. The original base definition does nothing, but glossaries-extra redefines this:

```
\renewcommand*{\glspostlinkhook}{%  
  \ifglsentryexists{\glslabel}{\glsxtrpostlinkhook}{}%  
}
```

This uses:

```
\glsxtrpostlinkhook
```

which is the main glossaries-extra post-link hook.

If you are migrating over from only using the base glossaries package and you have redefined `\glspostlinkhook`, consider moving your modifications to the category post-link hook or prepend to `\glsxtrpostlink`, as some attributes and abbreviation styles rely on the features provided by `\glsxtrpostlinkhook`.

The main post-link hook is defined as:
5. Referencing (Using) Entries

\newcommand*{\glsxtrpostlinkhook}{% \
glsxtrdiscardperiod{\glslabel}\
{\glsxtrpostlinkendsentence}\
{\glsxtrcustomdiscardperiod}\
{\glsxtrperiod{\glsxtrpostlinkendsentence}{\glsxtrpostlink}}\
{\glsxtrpostlink}}%
%
}

This checks if a following full stop needs to be discarded and does the inner post-link hook \glsxtrpostlinkhook. Note that \glsxtrdiscardperiod and \glsxtrperiod look ahead for a following token, so if you need to modify this command, insert your custom code at the start or add it to the category post-link hook instead.

\glsxtrdiscardperiod{⟨entry-label⟩}{⟨discarded⟩}{⟨no discard⟩}{⟨token⟩}

This discards ⟨token⟩ if it’s a full stop and the entry’s category attributes indicate that a full stop should be discarded (such as \gls discardperiod). If the punctuation character is discarded, this will then do ⟨discarded⟩, otherwise it will do ⟨no discard⟩ and process ⟨token⟩ as usual. If the \texttt{retainfirstuseperiod} attribute is set, then the following command is used to determine whether or not to discard ⟨token⟩.

\glsxtrdiscardperiodretainfirstuse{⟨entry-label⟩}{⟨discarded⟩}{⟨no discard⟩}{⟨token⟩}

This was introduced in v1.49 and is defined as:

\newcommand*{\glsxtrdiscardperiodretainfirstuse}[3]{% \
glsxtrifwassubsequentorshort{\glsxtrperiod{#2}{#3}{#3}}%
\}

This will only discard the full stop if it follows the subsequent use of a \gls-like command or if it follows one of the \glsxtrshort set of commands. Note that this has a different effect from pre v1.49 with the \glsxtrshort set of commands, but it’s more appropriate since it only the short form that requires the period to be discarded. To restore the original behaviour:

\renewcommand*{\glsxtrdiscardperiodretainfirstuse}[3]{% \
glsxtrifwasfirstuse{#3}{\glsxtrperiod{#2}{#3}{#3}}%
\}

If you want your own custom code to determine whether or not to check for a period
(instead of using known category attributes), you can redefine:

\texttt{\textbackslash{glsxtrifcustomdiscardperiod}\{\texttt{true}\}\{\texttt{false}\}\ initial: \ (false)}

This should expand to \texttt{\textlangle true\textrangle} if a check should be performed, otherwise it should expand to \texttt{\textlangle false\textrangle}. The default definition simply does \texttt{\textlangle false\textrangle}.

\texttt{\textbackslash{glsxtrpostlinkendsentence}}

This is done if a full stop is discarded. If there is a category post-link hook for the entry’s category, that hook is performed (\texttt{\textbackslash{glsxtrpostlink\langle category\textrangle}} not \texttt{\textbackslash{glsxtrpostlink}}) and the full stop is put back followed by a space factor adjustment. Otherwise, just the space factor adjustment is done.

The test to determine whether or not \texttt{\textlangle token\textrangle} is a full stop is determined by:

\texttt{\textbackslash{glsxtrifperiod}\{\texttt{true}\}\{\texttt{false}\}\langle token\rangle}

It may be useful to test for other punctuation characters. For example, styles such as \texttt{short-postfootnote} will move the footnote after certain punctuation characters.

\texttt{\textbackslash{glsxtrifnextpunc}\{\texttt{true}\}\{\texttt{false}\}}

This does \texttt{\textlangle true\textrangle} if it's followed by one of the set of recognised punctuation characters, otherwise it does false. The set is initialised to .,:;?! (full stop, comma, colon, semi-colon, question mark, and exclamation mark).

A convenient way of moving code after the punctuation character is to use:

\texttt{\textbackslash{glsxtrdopostpunc}\{\texttt{code}\}\langle token\rangle}

If \texttt{\textlangle token\textrangle} is a recognised punctuation character, this will place \texttt{\textlangle code\textrangle} after the token, otherwise it will be placed before the token.

The earlier example can be adapted to put the asterisk after following punctuation:

\texttt{\textbackslash{glsdefpostlink}\{general\}\{\textbackslash{glsxtrdopostpunc}\{\texttt{*}\}\}}
\texttt{\newglossaryentry\{sample\}\{name=\{sample\},symbol=\{X\},}
\texttt{description=\{an example\}\}}
\texttt{\begin{document}}

248
5. Referencing (Using) Entries

Example 102: Category post-link hook with punctuation lookahead

Sample,* sample,* (an example*) and X.*

Note that the asterisk isn’t moved after the closing parenthesis. This is because that character isn’t included in the default list.

You can add additional punctuation marks with:

\glsxtraddpunctuationmark{⟨token(s)⟩}

You may list multiple characters at the same time to add a batch, but don’t add any separators (including spaces).

Note that each character must be a single token, which means a single-byte character for pdfLaTeX. Multi-byte characters (UTF-8) will required a native Unicode engine (XeLaTeX or LuaLaTeX).

For example:

\glsxtraddpunctuationmark{-'/}

This adds three extra punctuation marks (hyphen, apostrophe and slash). Note that this doesn’t allow for closing double-quotes and will break "" (double apostrophe sequence for a closing double-quote) if found. The following will only work with XeLaTeX or LuaLaTeX:

\usepackage{fontspec}
\usepackage{glossaries-extra}
\glsxtraddpunctuationmark{"}

You can set the list with:

\glsxtrsetpunctuationmarks{⟨token list⟩}
This will remove the default set as well as any additional characters. As above, each character must be a single token with no separators in the list. For example:

\glsxtrsetpunctuationmarks{.?!}

This sets the list to just three punctuation characters (so comma, colon, and semi-colon are no longer recognised).

\glsxtrpostlink

This does the category post-link hook (or nothing if it hasn’t been defined):

\newcommand*{\glsxtrpostlink}{% 
  \csuse{glsxtrpostlink\glscategory{\glslabel}}%
}

Customisation is best performed within the category post-link hook, which can be defined (or redefined) with:

\glsdefpostlink{(category)}{(definition)}

The first argument is the category label and the second is the code to perform. Note that this doesn’t check if the hook has already been defined for the category. The hook is a command in the form \glsxtrpostlink{(category)}. If the category label only consists of letters, you can also use \newcommand or \renewcommand instead.

\glspretopostlink{(category)}{(code)}

Similar to the above but prepends \textit{(code)} to the associated hook (or simply defines it, if the hook doesn’t already exist).

\glsapptopostlink{(category)}{(code)}

Similar to the above but appends \textit{(code)} to the associated hook.

Take care not to choose category labels that will cause a conflict. For example, \textit{endsentence} and \textit{hook} will conflict with the commands \glsxtrpostlinkend-
If you want code in the post-link hook that’s not dependent on the category, consider prepending it to \glsxtrpostlink or \glsxtrpostlinkhook. Don’t append it to \glsxtrpostlinkhook otherwise it will interfere with the punctuation lookahead.

For convenience, some commands are provided that may be of use in the category post-link hook:

\glsxtrpostlinkAddDescOnFirstUse

This will add the description in parentheses if the hook follows the first use of the entry. This incorporates the inner formatting and description accessibility support, if provided.

\glsxtrpostlinkAddSymbolOnFirstUse

This will add the symbol in parentheses if that field is set and the hook follows the first use of the entry. This incorporates the inner formatting and symbol accessibility support, if provided.

\glsxtrpostlinkAddSymbolDescOnFirstUse

This will add the symbol, if that field is set, and the description (both within the same set of parentheses), if the hook follows the first use of the entry. This incorporates the inner formatting and accessibility support, if provided. The separator between the symbol and description is given by:

\glsxtrpostlinkSymbolDescSep

The default is a comma followed by a space.

For example:

\glsdefpostlink{general}{\glsxtrpostlinkAddDescOnFirstUse}
\glsdefpostlink{symbol}{\glsxtrpostlinkAddSymbolOnFirstUse}
\glsdefpostlink{number}{\glsxtrpostlinkAddSymbolDescOnFirstUse}
\newglossaryentry{sample}{name={sample},description={an example}}
\newglossaryentry{alpha}{name={alpha},symbol={\ensuremath{\alpha}},
description={a symbol},category={symbol}}
\newglossaryentry{pi}{name={pi},symbol={\ensuremath{\pi}},
5. Referencing (Using) Entries

\begin{document}
\begin{verbatim}
description={a constant}, category={number}
\begin{document}
First use: \gls{sample}, \gls{alpha}, \gls{pi}.

Next use: \gls{sample}, \gls{alpha}, \gls{pi}.
\end{document}
\end{verbatim}
\end{document}

This produces:

Example 103: Category post-link hooks

First use: sample (an example), alpha (α), pi (π, a constant).
Next use: sample, alpha, pi.

The following commands are also provided for use in the post-link hook:

\glsxtrcurrentfield

This expands to empty if the calling command isn’t associated with one specific field (such as \glslink, the \gls-like commands, the inline full form commands) otherwise it will expand to the name of the key associated with the singular form of the command. For example, this command will expand to text for both \glistext and \glspl, to description for both \glsdesc and \glsdescpl, and to short for both \glsxtrshort and \glsxtrshortpl. Whereas it will expand to nothing for both \gls and \glsxtrfull.

\glsxtrifwasglslike\{⟨true⟩\}{⟨false⟩}

This expands to ⟨true⟩ if the calling command was a \gls-like command and to ⟨false⟩ otherwise.

\glsxtrifwasglslikeandfirstuse\{⟨true⟩\}{⟨false⟩}

This expands to ⟨true⟩ if the calling command was a \gls-like command and was the first use otherwise it expands to ⟨false⟩. This is simply a shortcut command that uses both \glsxtrifwasglslike and \glsxtrifwasfirstuse.

\glsxtrifwassubsequentuse\{⟨true⟩\}{⟨false⟩}

This expands to ⟨true⟩ if the calling command was a \gls-like command and was the subsequent
use otherwise it expands to \langle false \rangle. This is simply a shortcut command that uses both \glxtrifwas glslike and \glxtrifwas firstuse.

\glxtrifwassubsequentorshort{\langle true \rangle}{\langle false \rangle}

This expands to \langle true \rangle if the calling command was a \gls-like command and was the subsequent use or if the calling command set \glsxtrcurrentfield to short. Otherwise it expands to \langle false \rangle.

\glxtrifallcaps{\langle all caps \rangle}{\langle not all caps \rangle}

This simply does:

\glscapscase{\langle not all caps \rangle}{\langle not all caps \rangle}{\langle all caps \rangle}

It’s not usually necessary for the post-link hook to differentiate between no case-change and sentence case, so this provides a convenient shortcut if only the all caps case needs to be different.

It’s possible you may also want to reference the inserted material. For the \gls-like commands, this can be obtained with the placeholder \glsinsert, but it’s not normally set by the \glstext-like commands, which don’t use the entry format style (§5.5.5) and instead incorporate the inserted material at the end of the link text. If you want the post-link hook to be able to access the inserted material for the \glstext-like commands, you must first save it, by redefining the following:

\glxtrsaveinsert{\langle entry-label \rangle}{\langle insert \rangle}

This is used by the \glstext-like commands to initialise \glsinsert. The default is:

\newcommand*{\glxtrsaveinsert}[2]{\def\glsinsert{}}

For example, to always save the insert:

\renewcommand*{\glxtrsaveinsert}[2]{\def\glsinsert{#2}}

The first argument can be used to conditionally assign the insert. For example, the following will only save it for entries with the general category:
5. Referencing (Using) Entries

\renewcommand*{\glsxtrsaveinsert}{%}
\glsxifcategory{#1}{general}{\def\glsinsert{#2}}{\def\glsinsert{}}%}

If you only want to save the insert for the \glsxtrfull set of commands, you can redefine \glsxtrfullsaveinsert instead (see §4.3).

\glsxtrassignlinktextfmt

This contains the assignments required to ensure that \glslabel, \glstextformat and \glsxtrgenentrytextfmt have the definitions they had within the link text. They would ordinarily still have those definitions within the post-link hook, but if, for example, the hook contains content that may be deferred, such as a footnote, then judicious use and expansion of \glsxtrassignlinktextfmt can allow the deferred code to pick up the label, outer and inner formatting.

For example, the post-link hook might contain:

\expandafter\footnote\expandafter
{\glsxtrassignlinktextfmt\glstextformat{%
\Glsaccessfmtdesc{}\{\glsxtrgenentrytextfmt\}\{\glslabel\}}.}

5.5.5. Entry Format Style

This section is for advanced users. Minor modifications can be made by adjusting the outer formatting (§5.5.1), the post-link hook (§5.5.4) or the abbreviation style commands (§4.5.1.3).

The \gls-like commands have the link text set to the entry format style corresponding to the entry’s glossary type. This can be changed with \def\glsentryfmt, but the default style is given by \glsentryfmt, which is defined by the base glossaries package. This uses the placeholder commands to determine the appropriate text. These are described in the glossaries manual, but to recap they are: \glslabel (the entry’s label), \glscustomtext (text provided by \glsdisp or empty otherwise), \glsinsert (supplied in the final optional argument except for \glsdisp, empty by default), \glsifplural, and \glscapscase.

The glossaries-extra package redefines \glsentryfmt to test whether or not the entry is an abbreviation and, if so, whether or not the entry should be treated as a regular entry:
5. Referencing (Using) Entries

\renewcommand*{\glsentryfmt}{%
  \ifglshasshort{\glslabel}{%\glssetabbrvfmt{\glscategory{\glslabel}}}{%}
  \glsifregular{\glslabel}{%\glsxtrregularfont{\glsgenentryfmt}}{%%
    \ifglshasshort{\glslabel}{%\glsxtrabbreviationfont{\glsxtrgenabbrvfmt}}{%\glsxtrregularfont{\glsgenentryfmt}}%
  }%
}%

This uses \ifglshasshort to determine whether or not the entry is an abbreviation. If it is, then \glssetabbrvfmt is used to setup the abbreviation style commands for the entry’s category.

Then there’s a check (with \glsifregular) to determine whether or not the entry should be treated as a regular entry. Note that if the regular attribute hasn’t been set to true, the entry will still be treated as a regular entry if it doesn’t have the short field set.

Regular entries are formatted according to:

\glsgenentryfmt

This is the generic regular entry format. It’s encapsulated with \glsxtrregularfont, but note that if the entry is an abbreviation, it will still use the abbreviation style formatting commands, which are contained within the first, firstplural, text and plural field values.

The generic regular entry format \glsgenentryfmt is provided by the base glossaries package, but is redefined by glossaries-extra to support inner formatting (§5.5.3) and accessibility (§9), if required.

Abbreviations that aren’t considered regular, are formatted according to:

\glsxtrgenabbrvfmt

This is the generic non-regular abbreviation format. It’s encapsulated with \glsxtrabbreviationfont. Unlike \glsgenentryfmt this doesn’t reference the first, firstplural, text or plural fields, but instead uses the abbreviation formatting commands \glsxtrfullformat, \glsxtrsubsequentfmt and their plural and case-changing variants.

If you want to define your own custom entry format, you will need to consider whether or not your format should support regular and non-regular abbreviation styles. Further detail can be found in the documented code:
5.6. Hyperlinks

The \gls-like and \glstext-like commands will automatically create a hyperlink by default, if \hyperref has been loaded (before \glossaries/glossaries-extra). The hyperlink can be switched off with \hyper=false but will also be switched off if the entry was assigned to an ignored glossary that was defined with the unstarred \newignoredglossary.

The \hyperfirst=false package option and the category attributes nohyper, nohyperfirst and nohypernext can also be used to automatically switch off the hyperlink. See also the \hyperoutside option that determines whether the hyperlink should be inside or outside of the outer formatting.

The hyperlink target is usually created by \glstarget which is used by all the predefined glossary styles by the standalone commands, such as \GlsXtrStandaloneEntryName.

The target for an entry with the label \langle entry-label \rangle is in the form \langle prefix \rangle\langle entry-label \rangle. The \langle prefix \rangle is normally \glolinkprefix but may be changed with the prefix option when displaying the glossary.

The target can also be changed to a link to an external file with the \targeturl category attribute.

5.7. Label Prefixes

It’s possible that you may want to prefix labels to ensure uniqueness. For example, this manual references both the \makeglossaries command and the makeglossaries Perl script. They are both defined as glossary entries, but they can’t both have the label makeglossaries. This manual uses bib2gls and is quite complicated, but a simplified version is as follows:

\newcommand{\csfmt}[1]{{\texttt{#1}}}
\newcommand{\appfmt}[1]{{\texttt{#1}}}
\newglossaryentry{cs.makeglossaries}{name={\csfmt{makeglossaries}},
  description={}}
\newglossaryentry{app.makeglossaries}{name={\appfmt{makeglossaries} },
  description={}}

So the label cs.makeglossaries refers to \makeglossaries and app.makeglossaries refers to makeglossaries. If you have a lot of prefixes like this, you may prefer to have a command that automatically adds the prefix. For example,
The problem with this is that the custom command \cs doesn’t allow for the *, + and \langle alt-mod\rangle modifiers (such as \gls* or \gls+). Instead you can use:

\glsxtrnewgls\{\langle default-options\rangle\}\{\langle prefix\rangle\}\{\langle cs\rangle\}

which defines the command

\langle cs\rangle\{\langle modifier\rangle\}\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\}

that behaves like

\gls\{\langle modifier\rangle\}\{\langle default options\rangle,\langle options\rangle\}\{\langle prefix\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\}

For example:

\glsxtrnewgls\{cs\}\{cs\}

or (to default to no hyperlinks)

\glsxtrnewgls\{hyper=false\}\{sym\}\{cs\}

now you can use \cs+{M} to behave like \gls+{cs.M}.

If you also want the plural and sentence case versions you can use

\glsxtrnewglslike\{\langle default-options\rangle\}\{\langle prefix\rangle\}\{\langle gls-like cs\rangle\}\{\langle glspl-like cs\rangle\}\{\langle gls-like cs\rangle\}\{\langle glspl-like cs\rangle\}

For example:

\glsxtrnewglslike\{hyper=false\}\{idx\}\{idx\}\{idxpl\}\{Idx\}\{Idxpl\}

For the all caps versions:

\glsxtrnewGLSlike\{\langle default-options\rangle\}\{\langle prefix\rangle\}\{\langle GLS-like cs\rangle\}\{\langle GLSpl-like cs\rangle\}
5. Referencing (Using) Entries

For example:

```
\glsxtrnewGLSlike[hyper=false]{idx.}{\IDX}{\IDXpl}
```

For commands that require the link text to be specified, you can use:

```
\glsxtrnewglslink[⟨default-options⟩]{⟨prefix⟩}{⟨cs⟩}
```

which defines ⟨cs⟩[⟨options⟩]{⟨label⟩}{⟨text⟩} to behave like \glslink[⟨default-options⟩,⟨options⟩]{⟨prefix⟩}{⟨label⟩}{⟨text⟩}, or

```
\glsxtrnewglsdisp[⟨default-options⟩]{⟨prefix⟩}{⟨cs⟩}
```

which defines ⟨cs⟩[⟨options⟩]{⟨label⟩}{⟨text⟩} to behave like \glsdisp[⟨default-options⟩,⟨options⟩]{⟨prefix⟩}{⟨label⟩}{⟨text⟩}.

If you are using \bib2gls, it can pick up the custom commands that are defined using the above, so it can detect dependencies when it parses fields such as description. If you provide your own custom command with just \newcommand that has syntax that starts with [⟨options⟩]{⟨entry-label⟩}, then you can notify \bib2gls using:

```
\glsxtridentifyglslike{⟨label-prefix⟩}{⟨cs⟩}
```

where ⟨label-prefix⟩ is the prefix to apply to the label that’s passed to the command ⟨cs⟩. The information is written to the aux file so that \bib2gls can add the given command to those it looks for when searching for dependencies.

Another possibility when using \bib2gls is to set up known label prefixes, see §11.5.7 for further details.

If you use \bib2gls with record counting, there are commands to \glsxtrnewgls for \rgls:

```
\glsxtrnewrgls[⟨default-options⟩]{⟨prefix⟩}{⟨cs⟩}
```

and for \rgls, \rglsp, \rGls and \rGlspl:

```
\glsxtrnewrglslike[⟨default-options⟩]{⟨prefix⟩}{⟨rgls-like cs⟩}{⟨rglsp-like cs⟩}{⟨rGls-like cs⟩}{⟨rGlsp-like cs⟩}
```

258
5. Referencing (Using) Entries

and for all caps:

\glsxtrnewGLSlike[\langle default-options\rangle]{\langle prefix\rangle}{\langle rGLS-like cs \rangle}{\langle rGLSpl-like cs \rangle}

Defining commands in this manner (rather than simply using \newcommand) also allows the command to be identified as a sentence case blocker to prevent the label from being converted or, in the case of \glsxtrnewglslike and \glsxtrnewrglslike, as a mapping. See §5.2 for further details.

5.8. Indexing

Indexing is normally performed implicitly by the \gls-like and \glstext-like commands, but this action can be prevented, such as by using the option noindex=true. These commands also generate text (the link text, §5.5). If you want to simply index an entry (to ensure that an entry is shown in the glossary) without producing any text then you can use \glsadd. Indexing is also performed by cross-referencing commands, such as \glssee. In the case of makeindex, \glssee simply behaves like \glsadd with a special format and the location set to Z (which pushes it to the end of the location list). Entries in ignored glossaries can only be indexed with bib2gls.

If you want all defined entries to appear in the glossary, regardless of whether or not they have been used in the document, then you can use \glsaddall or \glsaddallunused (both provided by the base glossaries package). These both iterate over all entries (in all non-ignored glossaries). In the first case (\glsaddall), every entry is indexed with the \glsadd options provided in the optional argument of \glsaddall. In the second case (\glsaddallunused), only those entries that haven’t been marked as used so far will be indexed using \glsadd[format=glsignore]{\langle label\rangle}. See the glossaries manual for further details of those commands.

The glossaries-extra package provides a similar command:

\glsaddallunindexed[\langle glossary types\rangle]

This is like \glsaddallunused but indexes all entries that haven’t been indexed so far (again using the option format=glsignore). This is preferable to \glsaddallunused if you have to reset the first use flag for any entries. As with \glsaddallunused, if this command is required, it should be placed near the end of the document. Indexing any entries after either of these commands are used will cause spurious commas in the location lists.
Iterative commands such as \glsaddall, \glsaddallunused and \glsaddallun-indexed should not be used with bib2gls. Use the selection=all option instead.

If you want to index a specific subset of entries, rather than all entries for a given glossary, you can use:

\glsaddeach\{⟨options⟩\}{⟨entry label list⟩}

This does \glsadd\{⟨options⟩\}{⟨entry-label⟩} for each entry in the comma-separated ⟨entry label list⟩. This command may be used with bib2gls, although it may be simpler to adjust the selection criteria or use filtering.

\GlsXtrAutoAddOnFormat\{⟨entry-label⟩\}{⟨format list⟩}{⟨glsadd options⟩}

This will make the \gls-like and \glstext-like commands automatically use \glsadd\{⟨gls-add options⟩\}{⟨entry-label⟩} whenever a \gls-like or \glstext-like command is used for the entry given by ⟨entry-label⟩ when the format matches one of the formats in the comma-separated ⟨format list⟩.

The optional argument ⟨label⟩ defaults to \glslabel (which will match ⟨entry-label⟩ that was used with \gls etc) and indicates the entry label to use in \glsadd and so needs to be expandable. The ⟨format list⟩ is a comma-separated list of format values that will trigger the automated adding. The ⟨glsadd options⟩ are the options to pass to \glsadd with format=⟨format⟩ prepended to the list.

For example, with:

\GlsXtrAutoAddOnFormat{hyperbf}{counter=chapter}

then \gls[format=hyperbf]{sample} will be equivalent to:

\glsadd[format=hyperbf,counter=chapter]{sample}\gls[format=hyperbf]{sample}

Note that the explicit range markers will prevent a match unless you include them in ⟨format list⟩ (in which case, be sure to add both the start and end formats).

Here’s another example:

\GlsXtrAutoAddOnFormat[dual.\glslabel]{hyperbf}{\}

In this case \gls[format=hyperbf]{sample} will now be equivalent to:
In the context of glossaries and glossaries-extra, indexing refers to the mechanism used to ensure that an entry is included in its associated glossary. (If you also want to use `\index`, see §12.) This includes any entries that simply cross-reference another entry. The default is to use `makeindex`, which is a general purpose indexing application. Each time an entry is indexed, a line is added to an associated file that contains the indexing information, which includes the sort value, the hierarchical information (if the entry has a parent) and an associated location (the page number, by default). This information is used to sort the entries and collate the locations into a compact location list. The `xindy` package option switches to using `xindy` syntax, but the process is much the same.

Since both `makeindex` and `xindy` are general purpose indexing applications they require an associated location (or a cross-reference) since indexes are typically used to lookup the locations in the document where the term occurs. Although glossaries are similar to indexes they can simply be used to provide brief summaries of each term without any locations. The way that `makeindex` and `xindy` work means that valid locations (that is, locations that conform to `makeindex`/`xindy` syntax) must be supplied even if no location list is required. If an invalid location is used, an error will occur during the `makeindex`/`xindy` step in the build process, even if the location will eventually be ignored when typesetting the glossary.

All location lists can be suppressed with the `nonumberlist` option (which simply discards the location list for each entry), but there are occasions where only some locations need to be suppressed. The main way of hiding a location is to encapsulate the location with a command that does nothing. The `\glsignore` command is used for this purpose (format=\glsignore). However, it’s important to remember that even though the location isn’t shown, it’s still present in the location list. This means that you will end up with spurious commas if there’s more than one item in the location list.

The “noidx” method similarly writes indexing information, but in this case the information is written to the `aux` file. Again, empty locations can cause spurious commas in the location lists.

The only method that recognises `\glsignore` as a special “ignored location” is `bib2gls`, where this format will trigger the entry’s selection but won’t add the ignored location to the location list. This avoids the problem of spurious commas caused by invisible locations.

The location corresponds to a counter. The default is the page counter, but may be changed with the `counter` package option, the `<counter>` optional argument of `\newglossary`, the `counter` key when defining an entry, or the `counter` option when indexing an entry.

Note that `bib2gls` v3.0+ converts an empty location (which can occur when the location counter is 0 and should be formatted as a Roman numeral) to an ignored location. For example, if you use `counter=part` but have `\gls` before the first `\part`. An empty location will
trigger an error with \texttt{makeindex} and \texttt{xindy}.

\begin{quote}
Since no entries are defined on the first \LaTeX\ run with \texttt{bib2gls}, there's no way of determining the entry's glossary \texttt{type} or of finding if the entry's \texttt{counter} key has been set. This means that if the counter has been assigned to either the entry's glossary or to the entry itself, the location counter can't be implemented until the entry has been defined. A second build is required to ensure that the locations use the correct counter.
\end{quote}

The location counter must expand to syntax that's recognised by the indexing application. This is very restrictive with \texttt{makeindex}, which only recognises Western Arabic (\texttt{\arabic}), lowercase Roman numerals (\texttt{\roman}), uppercase Roman numerals (\texttt{\Roman}), lowercase Basic Latin (\texttt{\alph}) and uppercase Basic Latin (\texttt{\Alph}), with optionally a separator (hyphen by default). With \texttt{xindy}, the syntax must be defined (see the \texttt{glossaries} manual for further details).

There's no restriction on the location syntax with \texttt{bib2gls}. The only limitation is that if \texttt{bib2gls} can't determine an associated numeric value according to its location parser, it won't form ranges. This means that with \texttt{bib2gls}, you can set arbitrary text as the location (that's not related to a counter) with \texttt{thevalue}. You can also use \texttt{thevalue} with \texttt{makeindex} and \texttt{xindy}, but only if the value matches the required location syntax.

Both \texttt{makeindex} and \texttt{xindy} order the locations in the location lists. For example:

\begin{verbatim}
\makeglossaries
\newglossaryentry{sample}{name={sample},description={an example}}
\begin{document}
\gls[thevalue=Z]{sample} (Z), \gls[thevalue=4]{sample} (4), \gls[thevalue=xi]{sample} (xi), \gls[thevalue=2]{sample} (2), \gls[thevalue=iii]{sample} (iii), \gls[thevalue=A]{sample} (A).
\end{document}
\end{verbatim}

Example 104: Location list ordering (\texttt{makeindex})

\begin{verbatim}
\gls[thevalue=Z]{sample} (Z), \gls[thevalue=4]{sample} (4), \gls[thevalue=xi]{sample} (xi), \gls[thevalue=2]{sample} (2), \gls[thevalue=iii]{sample} (iii), \gls[thevalue=A]{sample} (A).
\end{verbatim}

\section*{Glossary}

\begin{verbatim}
sample an example iii, xi, 2, 4, A, Z
\end{verbatim}

With \texttt{makeindex}, the location list is grouped into the different number formats (\texttt{\roman},...
5. Referencing (Using) Entries

\arabic and \Alph, with each group ordered numerically. The same result can be produced with \texttt{xindy} by adding the \texttt{xindy} package option to the above example.

With \texttt{bib2gls}, the location list is always in order of indexing. The above example document can be converted to use \texttt{bib2gls} as follows:

\begin{filecontents*}{\jobname.bib}
@entry{sample,name={sample},description={an example}}
\end{filecontents*}
\usepackage[record]{glossaries-extra}
\GlsXtrLoadResources
\begin{document}
\gls[thevalue=Z]{sample} (Z), \gls[thevalue=4]{sample} (4), \gls[thevalue=xi]{sample} (xi), \gls[thevalue=2]{sample} (2), \gls[thevalue=iii]{sample} (iii), \gls[thevalue=A]{sample} (A).
\end{document}

Example 105: Location list ordering (\texttt{bib2gls})

\begin{filecontents*}{\jobname.bib}
@entry{sample,name={sample},description={an example}}
\end{filecontents*}
\usepackage[record]{glossaries-extra}
\GlsXtrLoadResources
\begin{document}
\gls[thevalue=Z]{sample} (Z), \gls[thevalue=4]{sample} (4), \gls[thevalue=xi]{sample} (xi), \gls[thevalue=2]{sample} (2), \gls[thevalue=iii]{sample} (iii), \gls[thevalue=A]{sample} (A).
\end{document}

Glossary

\texttt{sample} an example Z, 4, xi, 2, iii, A

This example is contrived. For most documents, the order of indexing will likely match the desired location list order.

Another important difference between \texttt{bib2gls} and the other indexing methods is the treatment of cross-references identified by the cross-reference keys \texttt{see}, \texttt{seealso} and \texttt{alias}. With \texttt{bib2gls}, the cross-referencing information is picked up when \texttt{bib2gls} parses the \texttt{bib} file and is used to establish dependencies, which ensures that when entries with cross-references are selected, their cross-referenced entries will also be selected.

With the other methods, cross-references are added to an entry’s location list by indexing the entry with a special format. The \texttt{see}, \texttt{seealso} and \texttt{alias} keys automatically trigger this indexing unless \texttt{autoseeindex=false}. See §5.9 for further details.

Every time an entry is indexed, the following hook is also used:

\glsxtrdowrglossaryhook{(entry-label)}

This does nothing by default. The argument is the entry’s label.
The indexing code is encapsulated with:

\glsencapwrcontent{(code)}

This adds grouping, which helps to prevent spacing issues caused by the \texttt{whatsit} that’s created by the indexing.

The base \texttt{glossaries} package always performs the indexing before the link text for the \texttt{gls-like} and \texttt{glstext-like} commands. This means that if a page break occurs in the middle of the link text, the location will refer to the page number at the start of the link text (assuming the default page location counter). With \texttt{glossaries-extra}, you can use the option \texttt{wrgloss=after} to have the indexing occur after the link text. The \texttt{wrgloss} attribute can also be used. The default setting is initialised with \texttt{\glsxtrinitwrgloss} (see §5.1.1).

Every time an entry is indexed, an internal field associated with the entry’s label is globally updated to keep a count of the number of times the entry has been indexed. The value can be accessed with:

\glsentryindexcount{(entry-label)}

This command will expand to 0 if the entry hasn’t been indexed or hasn’t been defined. To test if the value is greater than 0 (that is, to test if the entry has been indexed yet), use:

\glsifindexed{(entry-label)}{(true)}{(false)}

This expands to \texttt{(true)} if the entry is defined and has been indexed, otherwise it expands to \texttt{(false)}. No warning or error occurs if the entry hasn’t been defined.

Note that the index count is a running total. This is not the same as the record count saved by \texttt{bib2gls}'s \texttt{--record-count} switch, which represents the total number of records for the given entry from the previous \LaTeX{} run.

The base \texttt{glossaries} package defines:

\glswriteentry{(entry-label)}{(code)}

This command conditionally writes the indexing code (supplied by the second argument \texttt{(code)}). The original definition simply tests whether or not the \texttt{indexonlyfirst} setting is on. The \texttt{glossaries-extra} package redefines this command to perform additional checks to determine whether or not the indexing code should be performed.
The modified definition uses:

\glsxtrifindexing\{⟨true⟩\}\{⟨false⟩\}

to test the \texttt{noindex} setting. This does \langle false \rangle if \texttt{noindex=true}, otherwise it does \langle true \rangle.

\ifglsindexonlyfirst \langle true \rangle \else \langle false \rangle \fi \texttt{initial: \texttt{iffalse}}

This is a conditional that corresponds to the \texttt{indexonlyfirst} package option. First use is tested using \GlsXtrIfUnusedOrUndefined rather than \texttt{ifglsused}. The \texttt{indexonly-first} attribute is also tested. If the “index only first” setting is on and the entry has been used, \langle code \rangle isn’t performed but auto-indexing via \texttt{\glsxtrdoautoindexname} is still performed (see §12).

Since no entries are defined on the first \LaTeX run with \texttt{bib2gls}, there’s no way of keeping track of whether or not an entry has been used or what its \texttt{category} is, which is required to query the \texttt{indexonlyfirst} attribute, so for the first document build all instances will be indexed. A second build is required for the “index only first” feature.

5.9. Cross-Referencing

The base \texttt{glossaries} package only provides the \texttt{see} key, which automatically indexes the cross-reference using \texttt{\glssee}. The value of this key isn’t saved and can’t be accessed later. (The key was simply provided as a shortcut.) The indexing ensures that the cross-reference is shown in the location list.

The auto-indexing feature of the \texttt{see} key was intended as a shortcut where only entries required in the document are defined. If you want to have a large file containing entries that may or may not be required in the document, then using \texttt{see, seealso or alias} can cause unwanted entries to appear in the glossary. In this case, see §5.9.1.

The \texttt{glossaries-extra} package saves the value of the \texttt{see} key and additionally provides the \texttt{seealso} and \texttt{alias} keys that perform similar functions. The values of the \texttt{see, seealso} and \texttt{alias} keys can all be accessed at a later point in the document.

If an entry with a cross-reference has been included in the glossary, there’s no guarantee that the cross-referenced entry will also be included. It won’t be included if it hasn’t been indexed anywhere in the document. You can use the \texttt{indexcrossrefs} package option to search for cross-references that require indexing at the end of the document, but note that this can be time-consuming if you have a large number of entries.
With bib2gls you can simply change the selection criteria (``selection={recorded
and deps and see}`` or `selection={recorded and deps and see not also}`) to
ensure that all cross-referenced entries are included even if they haven’t been
indexed in the document.

Example (``see`` and ``seealso`` keys):

```latex
\newglossaryentry{pumpkin}{name={pumpkin},description={}}
\newglossaryentry{cucumber}{name={cucumber},description={}}
\newglossaryentry{melon}{name={melon},description={}}
\newglossaryentry{gourd}{name={gourd},description={},
    see={pumpkin,cucumber,melon}}
\newglossaryentry{courgette}{name={courgette},description={}}
\newglossaryentry{marrow}{name={marrow},description={},
    seealso={courgette}}
```

When the gourd entry is defined, the cross-reference will automatically be indexed using `\glssee`. This means that the gourd entry will appear in the glossary, regardless of whether or not it is used in the document, with “see pumpkin, cucumber & melon” in the location list. If gourd is also indexed in the document, then those locations will also be added to the gourd’s location list.

The cross-referenced entries (pumpkin, cucumber and melon) will only appear in the glossary if they are also indexed in the document. This can be implemented automatically with `indexcrossrefs`.

The `seealso` key in the marrow entry functions in much the same way, but it is indexed with `\glsxtrindexseealso`. This means that the marrow entry will have “see also courgette” in its location list.

The `see` key may optionally start with `{tag}` to replace the default `\seename` tag with `{tag}. The `seealso` key doesn’t permit this. For example, the following is permitted:

```latex
\newglossaryentry{gourd}{name={gourd},description={},
    see={*[related topics]pumpkin,cucumber,melon}}
```

but you can’t replace `see` with `seealso` in the above as it would assume that the first label in the list is `[related topics]pumpkin` which is incorrect. The tag would have to be removed:

```latex
\newglossaryentry{gourd}{name={gourd},description={},
    seealso={pumpkin,cucumber,melon}}
```
5. Referencing (Using) Entries

(You could then redefine \seealso{name} to related topics, if required or redefine \glsxtrusealsoformat as applicable.)

Example (alias key):

\newglossaryentry{zucchini}{name={zucchini},description={}, alias={courgette}}

When the zucchini entry is defined, the alias key will automatically index zucchini with \glssee{zucchini}{courgette}. This means that the zucchini entry will be present in the glossary with “see courgette” in the location list. If the zucchini entry is referenced in the document using a command like \gls, then the hyperlink (if enabled) will go to the courgette entry (not the zucchini entry) but the zucchini entry won’t be indexed.

If you want the zucchini entry locations added to the courgette entry, you can redefine \glsxtrsetaliasnoindex (see §5.9.3) or, with bib2gls, use the alias-loc=transfer setting.

With bib2gls, cross-references are selected according to the selection criteria. See the bib2gls manual for further details.

5.9.1. Entries that may not be required

If you have a file containing a large number of entry definitions shared across multiple documents, then the use of the \seealso{name}, \seealso{name} or alias key can cause unwanted entries to appear in the document. This can be demonstrated as follows. Suppose the file myentries.tex contains:

\newglossaryentry{pumpkin}{name={pumpkin},description={}}
\newglossaryentry{cucumber}{name={cucumber},description={}}
\newglossaryentry{melon}{name={melon},description={}}
\newglossaryentry{gourd}{name={gourd},description={}, see={pumpkin,cucumber,melon}}
\newglossaryentry{cucurbit}{name={cucurbit},description={}, see={gourd}}
\newglossaryentry{courgette}{name={courgette},description={}}
\newglossaryentry{marrow}{name={marrow},description={}, seealso={courgette}}
\newglossaryentry{zucchini}{name={zucchini},description={}, alias={courgette}}
\newglossaryentry{broccoli}{name={broccoli},description={}}
Some of these entries have a cross-reference key set, but not all of these entries are required in the document:

\usepackage[colorlinks]{hyperref}
\usepackage[nostyles,stylemods=bookindex,style=bookindex]{glossaries-extra}
\makeglossaries
\loadglsentries{myentries}
\begin{document}
This document is only discussing \glspl{courgette} (baby \glspl{marrow}, also called a \gls{zucchini}), \glspl{pumpkin} and \glspl{melon}.
\printglossaries
\end{document}

Example 106: Cross-references (autoseeindex=true)

This document is only discussing courgettes (baby marrows, also called a zucchini), pumpkins and melons.

### Glossary

<table>
<thead>
<tr>
<th>C</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>cauliflower</td>
<td>marrow 1, see also courgette</td>
</tr>
<tr>
<td>courgette 1</td>
<td>melon 1</td>
</tr>
<tr>
<td>cucurbit see gourd</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>P</td>
</tr>
<tr>
<td>gourd see pumpkin, cucumber &amp; melon</td>
<td>pumpkin 1</td>
</tr>
<tr>
<td></td>
<td>Z</td>
</tr>
<tr>
<td></td>
<td>zucchini see courgette</td>
</tr>
</tbody>
</table>

Note that the glossary includes cucurbit and gourd, which aren’t referenced in the document. They could be useful as a redirect for the reader, but the gourd entry cross-references the cucumber entry, which isn’t included in the glossary, so the hyperlink target is undefined. The cauliflower entry has also been included in the glossary, but in this case it’s not useful.
for the reader as neither cauliflower nor broccoli (which it cross-references) are mentioned in the document. As with the cucumber cross-reference, the broccoli cross-reference hyperlink target is undefined.

There are a number of methods to address some of these problems. The first method has the cross-referencing keys in the tex file (as above), but disables the auto-indexing:

```
\usepackage[autoseeindex=false,nostyles,stylemods=bookindex, style=bookindex]{glossaries-extra}
\makeglossaries
\loadglsentries{myentries}
\begin{document}
This document is only discussing \gls{courgette} (baby \gls{marrow}, also called a \gls{zucchini}), \gls{pumpkin} and \gls{melon}.
\printglossaries
\end{document}
```

This doesn’t show the zucchini entry or any of the cross-references in the glossary because the information hasn’t been added to the indexing files. One way around this is to insert the cross-reference in a post-description hook.

```
\usepackage[autoseeindex=false,nostyles,stylemods=bookindex, style=bookindex]{glossaries-extra}
\makeglossaries
```

### Example 107: Cross-references (autoseeindex=false)

This document is only discussing courgettes (baby marrows, also called a zucchini), pumpkins and melons.

<table>
<thead>
<tr>
<th>Glossary</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>M</td>
</tr>
<tr>
<td>P</td>
</tr>
<tr>
<td>courgette</td>
</tr>
<tr>
<td>marrow 1</td>
</tr>
<tr>
<td>pumpkin 1</td>
</tr>
<tr>
<td>melon 1</td>
</tr>
</tbody>
</table>

```
5. Referencing (Using) Entries

\loadglsentries{myentries}
\glsdefpostdesc{general}{%
  \glsxtrseelists{\glscurrententrylabel}%
} 
\begin{document}
This document is only discussing \glspl{courgette} (baby \glspl{marrow},
also called a \gls{zucchini}), \glspl{pumpkin} and \glspl{melon}.
\printglossaries
\end{document}

Example 108: Cross-references (\texttt{autoseeindex=false} and post-name hook)
This document is only discussing courgettes (baby marrows, also called a zucchini), pumpkins and melons.

Glossary

\begin{tabular}{ll}
C & melon 1 \\
courgette 1 & \\
P & \\
M & marrow see also courgette 1 & pumpkin 1 \\
\end{tabular}

However, this still doesn’t solve the problem that the zucchini entry isn’t included in the glossary. It needs to be indexed, but indexing has been suppressed. Firstly, because the automatic indexing triggered by the \texttt{alias} key has been suppressed with \texttt{autoseeindex=false}, and, secondly, because the presence of the \texttt{alias} key automatically suppresses indexing with the \texttt{gls} and \texttt{glstext} like commands. This doesn’t cause a problem for the zucchini hyperlink, since the target is courgette (obtained from the \texttt{alias} key).

The second method is to not use those keys in the entry definitions and use \texttt{\glssee} or \texttt{\glsxtrindexseealso} within the document. For example, the file \texttt{myentries.tex} now contains:

\begin{verbatim}
\newglossaryentry{pumpkin}{name={pumpkin},description={}}
\newglossaryentry{cucumber}{name={cucumber},description={}}
\newglossaryentry{melon}{name={melon},description={}}
\newglossaryentry{gourd}{name={gourd},description={}}
\end{verbatim}
This document is only discussing \textit{\textsc{courgette}} (baby \textit{\textsc{marrow}}, also called a \textit{\textsc{zucchini}}), \textit{\textsc{pumpkin}} and \textit{\textsc{melon}}.

\printglossaries
Example 109: Cross-references (no see, seealso or alias)

This document is only discussing courgettes (baby marrows, also called a zucchini), pumpkins and melons.

Glossary

<table>
<thead>
<tr>
<th>C</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>courgette 1</td>
<td>pumpkin 1</td>
</tr>
<tr>
<td>gourd see</td>
<td>melon &amp;</td>
</tr>
<tr>
<td>courgette</td>
<td></td>
</tr>
<tr>
<td>marrow 1</td>
<td>zucchini see courgette</td>
</tr>
</tbody>
</table>

Note that aliases require the alias field to be set. In this case, I've set it with \GlsXtrSetField. The gourd and zucchini entries have been included in the glossary because they were added with \glssee. The other entries are in the glossary because they were indexed when referenced with \gls or \glspl.

Since cucumber isn’t required in the document, I haven’t included it in the cross-reference list for gourd. This method is flexible as it allows the cross-referencing to vary between documents. For example, another document may instead have:

\glsxtrindexseealso{pumpkin}{courgette,melon}
\glsxtrindexseealso{melon}{pumpkin,courgette}
\glsxtrindexseealso{courgette}{pumpkin,melon}

The third method is to switch to bib2gls. The file myentries.tex can be converted to myentries.bib using:

convertgls2bib --index-conversion myentries.tex myentries.bib

I’ve used the option --index-conversion (or -i) which will use @index instead of @entry for entries that have an empty description (which is the case in this example). This creates the file myentries.bib, which contains the following (space compacted):
The earlier example 106 on page 268 can be rewritten as:

```latex
\usepackage[colorlinks]{hyperref}
\usepackage[record,nostyles,stylemods=bookindex,style=bookindex]{glossaries-extra}
\GlsXtrLoadResources[src=myentries]
\begin{document}
This document is only discussing \gls{courgette} (baby \gls{marrow}, also called a \gls{zucchini}), \gls{pumpkin} and \gls{melon}.
\printunsrtglossaries
\end{document}
```

In order to support letter groups, \texttt{bib2gls} needs to be invoked with the --group switch.

The result is:
5. Referencing (Using) Entries

Example 110: Cross-references (bib2gls)

This document is only discussing courgettes (baby marrows, also called a zucchini), pumpkins and melons.

Glossary

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>P</td>
</tr>
<tr>
<td>courgette, 1</td>
<td>pumpkin, 1</td>
</tr>
<tr>
<td>M</td>
<td>Z</td>
</tr>
<tr>
<td>marrow, 1, see also courgette</td>
<td>zucchini, see courgette</td>
</tr>
</tbody>
</table>

This uses the default selection={recorded and deps}, which selects entries that have records, and their dependencies. Records correspond to the usual indexing performed by the \gls-like, \glstext-like or \glsadd commands. With bib2gls, the cross-referencing fields don’t trigger an index but identify dependencies.

Note that the above doesn’t include the gourd entry (which cross-references entries that have been indexed). The selection criteria can be changed to also include unrecorded entries that cross-reference selected entries. There are two options to choose from: selection={recorded and deps and see}, which will apply to all cross-reference fields (see, see-also and alias), or selection={recorded and deps and see not also}, which doesn’t consider the seealso field.

Changing the resource options in the above example to:

\GlsXtrLoadResources[src=myentries, selection={recorded and deps and see}]

results in:
Example 111: Cross-references (\texttt{bib2gls} and \texttt{selection=recorded and deps and see})

This document is only discussing \textit{courgettes} (baby marrows, also called a zucchini), pumpkins and melons.

\textbf{Glossary}

\begin{center}
\begin{tabular}{ll}
\textbf{C} & \textbf{M} \\
courgette, 1 & marrow, 1, \textit{see also} courgette \\
cucumber & melon, 1 \\
cucurbit, \textit{see} gourd & \\
\textbf{G} & \textbf{P} \\
gourd, \textit{see} pumpkin, cucumber & pumpkin, 1 \\
& \textit{melon} \\
\textbf{Z} & \\
zucchini, \textit{see} courgette &
\end{tabular}
\end{center}

This now includes the gourd entry because it cross-references pumpkin and melon, which have been recorded in the document. The cucurbit entry is also included because it cross-references the (now selected) gourd entry. Note that the cucumber entry has been selected because the gourd entry depends on it. This means there are no broken links in the glossary, but it looks a bit odd as the cucumber entry has no location list. As from \texttt{bib2gls v3.0}, this can be removed with one of the cross-reference pruning options, such as \texttt{prune-xr}. For example:

\begin{verbatim}
\texttt{\GlsXtrLoadResources[src=myentries,selection={recorded and deps and see},prune-xr]}
\end{verbatim}

results in:
5. Referencing (Using) Entries

Example 112: Cross-references (bib2gls and selection=recorded and deps and see, prune=xr)

This document is only discussing courgettes (baby marrows, also called a zucchini), pumpkins and melons.

Glossary

C  melon, 1

courgette, 1
cucurbit, see gourd

P

gourd, see pumpkin & melon

G  pumpkin, 1

M

marrow, 1, see also courgette

Z  zucchini, see courgette

This has removed the unnecessary cucumber from the gourd’s see list, and so cucumber doesn’t get selected.

See the bib2gls user manual for further details on the cross-reference selection and pruning options.

5.9.2. Accessing the Cross-Referencing Fields

If you have switched off the indexing of the cross-reference fields (with autoseeindex=false) or want to suppress the location lists, then you can adjust the glossary style or hooks to include the cross-references since they won’t be shown otherwise.

\glsxtrseelists{⟨entry-label⟩}

If the entry given by ⟨entry-label⟩ has the see,seealso or alias fields set, this will display the cross reference according to \glsxtruseseeformat (for see and alias) or \glsxtruseseealsoformat (for seealso). If any of these fields are set, the list is encapsulated with:

\glsxtrseelistsencap{⟨content⟩}
This simply does a space followed by \langle content \rangle. If more than one of the fields are set (not recommended), then they will be displayed in the order: see, seealso and alias. The entire set will be encapsulated with \glsxtrseelistsencap and each sub-list will be separated with:

\glsxtrseelistsdelim

which defaults to a comma followed by a space.

\glsxtrusesee{⟨entry-label⟩}

If the entry given by ⟨entry-label⟩ has the see field set, this will display the cross reference according to \glsxtruseseeformat, otherwise this does nothing. An error (or warning with undefaction=warn) will occur if the entry hasn’t been defined.

\glsxtrusealias{⟨entry-label⟩}

As \glsxtrusesee but for the alias field.

\glsxtrusealso{⟨entry-label⟩}

If the entry given by ⟨entry-label⟩ has the seealso field set, this will display the cross reference according to \glsxtrusealsoformat, otherwise this does nothing. An error (or warning with undefaction=warn) will occur if the entry hasn’t been defined.

\glsxtralias{⟨entry-label⟩}

This expands to the value of the alias field (which should be a single entry label) or empty if the field isn’t set. If the entry isn’t defined, this command will expand to \relax (without any error or warning). If you want to first test if the field is set, you can use \glsxtrifhas-field.

\glsxtrseealsolabels{⟨entry-label⟩}

This expands to the value of the seealso field (which should be a comma-separated list of entry labels) or empty if the field isn’t set. If the entry isn’t defined, this command will expand to \relax (without any error or warning). If you want to first test if the field is set,
you can use \glsxtrifhasfield.

\begin{verbatim}
\glsxtruseseealsoformat{(csv-list)}
\end{verbatim}

This command is used to format a “see also” cross-reference. This is simply defined to do:

\begin{verbatim}
\glsseeformat[\seealsoname]{(csv-list)}{()}
\end{verbatim}

### 5.9.3. Cross-Reference Indexing

If you are using bib2gls, see the bib2gls user manual for information about the selection, see, seealso, alias, alias-loc options.

The actual indexing of the seealso key is performed with:

\begin{verbatim}
\glsxtrindexseealso{(entry-label)}{(xr-list)}
\end{verbatim}

which is analogous to \glssee. As with \glssee, this can also be used explicitly.

With makeindex, \glsxtrindexseealso simply does:

\begin{verbatim}
\glssee[\seealsoname]{(entry-label)}{(xr-list)}
\end{verbatim}

With xindy, \glsxtrindexseealso behaves in an analogous way, using the appropriate cross-referencing markup.

\begin{verbatim}
\glsxtrsetaliasnoindex
\end{verbatim}

This hook is used within the \gls-like and \glstext-like commands to automatically switch off the indexing for aliases. (The hook is performed after the options set by \GlsXtrSetDefaultGlsOpts.)

By default, this hook just sets noindex=true. If you would like to add locations to the aliased location list then you can redefine it to use:

\begin{verbatim}
\glsxtrindexaliased
\end{verbatim}

For example:
5. Referencing (Using) Entries

\renewcommand{\glsxtrsetaliasnoindex}{\glsxtrindexaliased}

Note that this needs noindex=false to ensure the indexing takes place so don’t simply append \glsxtrindexaliased to the definition of \glsxtrsetaliasnoindex.

Don’t use the above hooks with bib2gls as this function is disabled with record=only and record=nameref. Use the alias-loc resource option instead.

\glsxtraddallcrossrefs

This is used at the end of the document if indexcrossrefs=true to automatically index any cross-references (identified in the see, seealso and alias fields). This command iterates over all entries in all glossaries and, if an entry has been marked as used, does:

\glsxtraddunusedxrefs

which indexes any labels identified in the cross-reference fields of the entry given by ⟨entry-label⟩ that haven’t been marked as used.

This can be time consuming if there are a large number of entries defined. If this is the case, you may want to consider switching to bib2gls and use either selection={recorded and deps and see} or selection={recorded and deps and see not also}.

There should be no need to use \glsxtraddallcrossrefs explicitly, but you may want to redefine it to only iterate over specific glossaries. The unused entries are indexed using the \glsxtrunusedformat format.

\glsxtrunusedformat{⟨location⟩}

This ignores its argument (the location) and just does \unskip.

5.10. First Use Flag

Each entry has an associated first use flag (a conditional or boolean variable), which determines whether or not the entry has been marked as “used”. Unsetting this flag means that the entry is marked as used. Resetting the flag means that the entry is marked as unused.

The \gls-like commands (which are the principle method of referencing an entry) all mark the entry as used after the link text is displayed but before the post-link hook is used.
The purpose of this is to allow for additional information that needs to be shown when a term first appears in a document. For example, an abbreviation may need to have its full form shown on the instance. However, there are some cases where that additional information may need to be shown again or where the literal first instance of the term may need to be in its terse form. For example, if the term is used in the front matter.

If any \gls-like commands (which are robust) are used in section headings or captions, they can end up in the table of contents or corresponding “list of...” (such as the list of figures). This can cause the first use flag to be unset too soon. For these situations, use the commands described in §5.3 instead.

The base glossaries package provides commands to explicitly unset or reset the first use flag either locally (confined to the current scope) or globally. These commands are: \glsunset (global unset), \glslocalunset (local unset), \glsreset (global reset) and \glslocalreset (local reset).

The glossaries-extra package adds hooks to the above commands. These do nothing by default, but are modified by \glsenableentrycount and \glsenableentryunitcount to perform the count increment or reset (see §6.1).

\glsxtrpostunset{⟨entry-label⟩}
This hook is added to \glsunset.

\glsxtrpostlocalunset{⟨entry-label⟩}
This hook is added to \glslocalunset.

\glsxtrpostreset{⟨entry-label⟩}
This hook is added to \glsreset.

\glsxtrpostlocalreset{⟨entry-label⟩}
This hook is added to \glslocalreset.

The base package also provides commands to unset or reset all entries or all entries within particular glossaries: \glsunsetall and \glsresetall. For example, if you don’t want the first use in the front matter, you can unset all entries at the start of the front matter and reset them at the start of the main matter.
With glossaries-extra you can unset a specific subset of entries.

\glslocalunseteach{⟨entry-labels⟩}

Locally unsets each entry in the given comma-separated list of entry labels.

\glslocalreseteach{⟨entry-labels⟩}

Locally resets each entry in the given comma-separated list of entry labels.

You can test if an entry has been marked as used with \ifglsused (but take care if you are using bib2gls or the undefaction=warn option, see below). This command allows the entry display style to vary the link text according to whether or not the entry has been marked as used. However, it can’t be used within the post-link hook as by that time, the first use flag will have already been unset.

See the glossaries user manual for further details of the above commands.

For example, in the following document the “html” entry is first used in the abstract, which shows both the long and the short form, but it would be helpful for the full form to be reshown in the main section about web pages. This is achieved by resetting the first use flag.

\newabbreviation{html}{HTML}{hypertext markup language}
\begin{document}
\begin{abstract}
This abstract mentions \gls{html}.
\end{abstract}
Some casual reference to \gls{html}.
\section{Web Pages}
\glsreset{html}This section is all about \gls{html}.
\end{document}
5. Referencing (Using) Entries

Example 113: Resetting the first use flag (\glsreset)

Abstract

This abstract mentions hypertext markup language (HTML).

Some casual reference to HTML.

1 Web Pages

This section is all about hypertext markup language (HTML).

In the above example, an alternative is to use \glsxtrfull where you particularly want the full form, but some abbreviation styles have a different expansion with the inline \glsxtrfull form compared with the first use of \gls.

The glossaries-extra package provides the options preunset and prerest, which can be used to unset or reset the first use flag before the link text. This means that in the above example, the line:

\glsreset{html}This section is all about \gls{html}.

can be replaced with:

This section is all about \gls[prereset]{html}.

As mentioned above, the first use flag is unset before the post-link hook, so \ifglsused isn’t helpful in the post-link hook. Instead, you can use:

\glsxtrifwasfirstuse{{true}}{{false}}

This command is initialised by the \gls-like commands according to the value of the first use flag before the link text. It’s also initialised by the \glstext-like commands: not according to the value of the first use flag but according to whether or not the \glstext-like command emulates first use.

For example, \gls will define \glsxtrifwasfirstuse to do its first argument if the first use flag indicates the entry hasn’t yet been used, otherwise it will define \glsxtrifwasfirstuse to do its second argument. Whereas \glsfirst will always define \glsxtrifwasfirstuse to do its first argument (unless used with preunset) and \glstext will always define \glsxtrifwasfirstuse to do its second argument (unless used with prereset), regardless of the state of the first use flag.
The `preunset` and `prereset` options will additionally redefine `\glsxtrifwasfirstuse` to match the option. See §5.5.4 for further details about the post-link hook.

If you want to check if the calling command was both the first use and it was a `\gls`-like command, you can use: `\glsxtrifwasglslikeandfirstuse`.

The unset function performed by the `\gls`-like commands before the post-link hook uses the global `\glsunset` by default. If you want `\glslocalunset` instead, you can use the `local` option (provided by the base glossaries package) or `postunset=local`. To prevent the first use flag from being unset after the link text, use `postunset=none`.

\begin{document}
{\% local scope
`\gls[local]{html}. Used? `\ifglsused{html}{Yes}{No}.\}
}% end scope

Used? `\ifglsused{html}{Yes}{No}.\}

`\gls[postunset=none]{html}. Used? `\ifglsused{html}{Yes}{No}.\}
\end{document}

Example 114: Local unset

hypertext markup language (HTML). Used? Yes.
Used? No.

If you are using the `undefaction=warn` option (which is automatically implemented by the `record` option), the first use flag is undefined and so is neither true nor false, in which case `\ifglsused` will trigger an error or warning and do neither. In this situation, you may need to use the following command instead.

\begin{verbatim}
\GlsXtrIfUnusedOrUndefined{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
\end{verbatim}

This does ⟨true⟩ if the entry hasn’t been defined or hasn’t been marked as used, otherwise does ⟨true⟩. Note that this command will generate an error or warning (according to `undefaction`) if the entry hasn’t been defined, but will still do ⟨true⟩. This is more useful than `\ifglsused` with `bib2gls` where the entries are never defined on the first \LaTeX run.

283
5.10.1. Buffering Unsets

Sometimes commands like \gls are used in a context where changing a boolean variable can cause things to go wrong. The outer, middle and inner formatting (see §5.5) can be used to change the font for the link text, but it may be that the \gls-like command occurs within a block of text that needs to be encapsulated by such a command.

One example of this is using \gls in one of the commands provided with the soul package. For example:

\begin{verbatim}
\ul{Some text about \gls{html}.}
\end{verbatim}

This causes the confusing error:

Glossary entry `{html}' has not been defined.

The simplest workaround is to put \gls{html} inside the argument of \mbox. For example:

\begin{verbatim}
\ul{Some text about \mbox{\gls{html}}.}
\end{verbatim}

This can work provided it’s not the first use of this entry. If it is, then unsetting the first use flag causes a problem and results in the error:

! Package soul Error: Reconstruction failed.

The glossaries-extra package provides a way of temporarily switching off \glsunset so that it just makes a note of the entry’s label but doesn’t actually perform the change.

\begin{verbatim}
\GlsXtrStartUnsetBuffering
\end{verbatim}

This starts the buffering. The unstarrred version doesn’t check for duplicates, so the internal list may end up with multiple occurrences of the same label. The starred version only adds a label to the internal list if it’s not already in it. If you are using entry counting (see §6.1) the unstarrred version is preferable to ensure the entry count is correct.

Buffering only applies to the global \glsunset and does not affect the local \glslocalunset.
5. Referencing (Using) Entries

The buffer can be locally cleared with:

\GlsXtrClearUnsetBuffer

This doesn’t stop buffering. It will simply discard the labels that have been buffered so far.

In order to restore the normal behaviour of \glsunset, the buffering must be stopped or discarded.

\GlsXtrStopUnsetBuffering \textit{modifier: *}

This stops the buffering, restores \glsunset, and unsets all the buffered labels. The starred form uses \glslocalunset to unset the buffered labels.

Before you stop the unset buffering, you can iterate over the current buffer.

\GlsXtrForUnsetBufferedList\{\langle handler-cs\rangle\}

This iterates over the currently buffered list of entry labels and performs \langle handler-cs\rangle\{\langle entry-label\rangle\} for each label, where \langle cs \rangle is a control sequence that takes a single argument. This is best used with the starred version of \GlsXtrStartUnsetBuffering* to avoid duplicates.

\GlsXtrDiscardUnsetBuffering

This discards the buffer and restores \glsunset to its normal behaviour.

It’s possible to locally unset entries before use (analogous to preunset=local) if the entry has already been encountered in the buffer. This will still be problematic for situations where changing a conditional causes a problem, but may be useful in some situations. This feature is enabled with:

\GlsXtrUnsetBufferEnableRepeatLocal

This may be placed before or after \GlsXtrStartUnsetBuffering but the locally collected list of unused entries will be cleared at the start of each instance of \GlsXtrStartUnsetBuffering. It will also be cleared by \GlsXtrClearUnsetBuffer. All entries that have been marked as unused can be reset with:

\GlsXtrResetLocalBuffer
5. Referencing (Using) Entries

This will perform a local reset on all the entries in the “not used” list and do \GlsXtrClear-UnsetBuffer.

This feature can be switched off with:

\GlsXtrUnsetBufferDisableRepeatLocal

It’s disabled by default.

The way this feature works is as follows (while buffering is active):

1. Each time an entry is referenced with a \gls-like command, the \glsinitreunsets hook checks if the current entry (identified by \glslabel) has been added to the buffer. (Bear in mind that the label is added to the buffer after the link text when \glsunset is used.)
   a) If it has been added to the buffer, then this is an indication that the entry has already been used within the buffer zone (that is, an attempt has been made to globally unset the first use flag). A local unset is then performed, which is essentially equivalent to using the \preunset=local option, so the reference will behave like subsequent use.
   b) If it hasn’t been added to the buffer, then this is an indication that the entry hasn’t already been used within the buffer zone, but it may or may not have been used before the buffering started. If the first use flag indicates that the entry hasn’t been used, the entry’s label will be added to the “not used” list. The reference will behave like first use, but the unset won’t be performed afterwards (because buffering is in progress).

2. The entries that are in the “not used” list can be locally reset and both the buffer and the “not used” list can be cleared with \GlsXtrClearUnsetBuffer.

Note that this approach can’t be used for situations where the change in conditional causes a problem, but it can be used in situations where the content of an environment or command is repeatedly processed, which upsets the first use flag.

For example, consider the following beamer document:

\documentclass{beamer}
\usepackage{glossaries-extra}
\newabbreviation{svm}{SVM}{support vector machine}
\newabbreviation{html}{HTML}{hypertext markup language}
\begin{document}
\begin{frame}
\frametitle{Frame 1}
\begin{itemize}
\item \gls{html} and \gls{html}
\end{itemize}
\end{frame}
\end{document}
The first page isn’t a problem as the frame doesn’t have overlays. The first reference of the “html” entry shows the full form and the next shows just the short form. The second page (which is the first of the overlays of the second frame) correctly shows the full form of the “svm” entry for the first reference and the short form for the second reference, but on the third page (the second of the overlays) now has all instances of “svm” showing as subsequent use (just the short form).

I could put \glslocalresetall at the start of the frame, but that would reset the “html” entry as well. Another workaround is to locally reset the first “svm” entry with \preresetlocal, but that defeats the point of the first use flag, which is intended to keep track of whether or not you have used an entry so that you don’t have to.

The frame can be placed inside a buffering zone:

\GlsXtrStartUnsetBuffering
\begin{frame}
  \frametitle{Frame 2}
  \begin{itemize}
  \item<+-> \gls{svm} and \gls{svm}
  \item<+-> \gls{svm} and \gls{html}
  \end{itemize}
\end{frame}
\GlsXtrStopUnsetBuffering

This ensures that the first use flag isn’t reset until after the frame, but it means that all references to the “svm” entry on both the second and third page show the full form.

The “repeat local” function can be used so that repeated references for the same entry can be locally unset before use. This can be enabled with \GlsXtrUnsetBufferEnableRepeatLocal which fixes the second page, but not the third page, which shows all references to “svm” as the short form. What’s needed is to locally reset and entries that are in the frame but haven’t yet been used (“svm”, in this case) at the start of the frame with \GlsXtrResetLocalBuffer:
Note that on the first overlay, the buffer and “not used” list are both empty. On the second overlay, the buffer contains the “svm” and “html” labels and the “not used” list just contains the “svm” label. The reset performed by \GlsXtrResetLocalBuffer will reset “svm” and then clear both the buffer and the “not used” list. This means that the first “svm” reference is once again considered first use and it will once again be added to the “not used” list (so that it would be reset again if there was a third overlay).

Example 115: Abbreviations with beamer (unset buffering)

This is quite cumbersome, but these commands could potentially be added to hooks at the start and end of problematic environments (but the buffering needs to be started and ended outside of the repeated content).

The following example uses \mbox to protect \gls within the buffer zone:
5. Referencing (Using) Entries

This produces:

Example 116: Buffering first use unsets with `mbox`

Some text about hypertext markup language (HTML). Next use: hypertext markup language
Next use: HTML.

Note that the use of `mbox` prevents line-breaking and the second instance of `\gls{html}` is treated as first use.

Note that since the change in the first use flag now doesn’t occur until `\GlsXtrStopUnsetBuffering`, multiple references of the same term within the buffering zone will always be treated as first use (if the term wasn’t used before the buffering started).

Other alternatives include using `\protect` and inner formatting (see §5.5.3 for limitations) or middle formatting (see §5.5.2) with `\GlsXtrExpandedFmt` (which can’t be used with fragile link text). Both approaches are demonstrated in the following example:
5. Referencing (Using) Entries

\renewcommand{\glsxtrdefaultentrytextfmt}[1]{\ul{#1}}%
\ul{Some text about \protect\gls{html}.}
Next use: \protect\gls{html}{}
}
Next use: \gls{html}.

Second approach (middle formatting with expanded link text):
\glsresetall
{% scope
\renewcommand{\glsxtrabbreviationfont}[1]{\xpul{#1}}%
\renewcommand{\glsxtrregularfont}[1]{\xpul{#1}}%
\ul{Some text about \protect\gls{html}.}
Next use: \gls{html}.
}
Next use: \gls{html}.
\end{document}

This produces:

Example 117: Alternatives to buffering

  First approach (inner formatting): Some text about hypertext markup language (HTML). Next use: HTML
Next use: HTML.
  Second approach (middle formatting with expanded link text): Some text about hypertext markup language (HTML). Next use: HTML.
Next use: HTML.

The change in the first use flag isn’t the only content within the \gls-like commands that can cause a problem. The whatsit caused by indexing can also be problematic. Buffering can also be used to help with that situation. Indexing can be switched off at the start of the buffering and \GlsXtrForUnsetBufferedList can be used to perform the indexing outside of the problematic content. Note that this can cause a problem if the location changes (for example, if a page break occurs within the buffering zone).

Buffering can also be used to simply gather the labels that have been referenced with a \gls-like command in order to, for example, display a mini-glossary at the end of the block. See for example, Gallery: Mini-Glossary.¹

¹dickimaw-books.com/gallery/index.php?label=minigloss
5.11. Accessing Fields

See §3.5 for setting fields after an entry has been defined, §5.13 for fields that contain comma-separated lists or whose values may be contained within comma-separated lists, §5.9.2 for cross-referencing fields (see, seealso and alias), and §5.15 for testing field values. See also the base glossaries package’s commands, such as \glsentryname and \glsletentryfield.

\glsxtrusefield{⟨entry-label⟩}{⟨field-label⟩}

This expands to the value of the field (identified by its internal label ⟨field-label⟩) for the entry identified by ⟨entry-label⟩. Expands to \relax if the field or entry are undefined.

\Glsxtrusefield{⟨entry-label⟩}{⟨field-label⟩}

This is like \glsxtrusefield but converts the first character to uppercase using \makefirstuc (provided by mfirstuc) which is robust. If hyperref is loaded, \Glsxtrusefield{⟨entry-label⟩} will use the expandable:

\MFUsentencecase{\glsxtrusefield{⟨entry-label⟩}}

in a PDF bookmark.

\GLSxtrusefield{⟨entry-label⟩}{⟨field-label⟩}

This is like \glsxtrusefield but converts the field value to uppercase. See §5.2.3.

\glsxtrfieldtitlecase{⟨entry-label⟩}{⟨field-label⟩}

This is like \glsxtrusefield but converts the field value to title case. This internally uses:

\glsxtrfieldtitlecases{⟨content⟩}

This converts ⟨content⟩ to title case (expanding the first token once). If \glscapitalisewords has been defined, that will be used, otherwise \capitalisewords will be used.

\glsxtrentryparentname{⟨entry-name⟩}
5. Referencing (Using) Entries

Expands to the entry’s parent name if defined. Expands to nothing if the entry doesn’t have a parent or if the entry isn’t defined. If you simply require the parent label then use \glsentryparent or, to first test if the entry has a parent, either use \ifglsasparent or use \glsxtrifhasfield with the field label set to parent.

\glsxtrhiername{⟨entry-label⟩}

Displays the hierarchical name for the given entry. The cross-reference format \glsseeitemformat may be redefined to use this command to show the hierarchy, if required.

This command has a recursive definition. If the entry given by ⟨entry-label⟩ has a parent, then this command will do \glsxtrhiername{⟨parent-label⟩} for the entry’s parent and will then do the separator \glsxtrhiernamesep.

Then, regardless of whether or not the entry has a parent, it will do \glsfmttext{⟨entry-label⟩}, if the entry is an abbreviation (see §1.2.4), or \glsfmtname{⟨entry-label⟩} otherwise.

If hyperref is loaded, \glsxtrhiername will behave as \glsentryname in a PDF bookmark.

\glsxtrhiernamesep

Separator symbol (⊿) used between each name in commands like \glsxtrhiername.

\Glsxtrhiername{⟨entry-label⟩}

As \glsxtrhiername but the first name in the list has its first character converted to uppercase using \Glsfmttext or \Glsfmtname (sentence case). If hyperref is loaded, \Glsxtrhiername will expand to:

\MFUsentencecase{\glsentryname{⟨entry-label⟩}}

in a PDF bookmark. The \makefirstuc mapping from \glsxtrhiername to \Glsxtrhiername is set with \glsmfuaddmap, if supported.

\GlsXtrhiername{⟨entry-label⟩}

As \glsxtrhiername but each name in the list has its first character converted to uppercase
5. Referencing (Using) Entries

using \Glsfmttext or \Glsfmtname.

\GLSxtrhiername{⟨entry-label⟩}

As \glsxtrhiername but the first name in the list is converted to uppercase using \GLSfmttext or \Glsfmtname.

\GLSXTRhiername{⟨entry-label⟩}

As \glsxtrhiername but each name in the list is converted to uppercase using \GLSfmttext or \Glsfmtname (all caps).

5.12. Encapsulation (Formatting) Based on Field Values

These commands assume that a given entry has a special purpose field that’s used to store information on how to format text.

5.12.1. Foreign Language Field

\GlsXtrForeignTextField

This command should expand to the internal field label used to store a language tag (such as en-GB or de-CH-1996). The default value is userii (which corresponds to the user2 key).

\GlsXtrForeignText{⟨entry-label⟩}{⟨text⟩}

If the entry given by ⟨entry-label⟩ has the field identified by \GlsXtrForeignTextField set, then this command will encapsulate ⟨text⟩ according to the language tag stored in that field.

This uses tracklang’s interface to determine the language label that corresponds to the language tag. If the language label can be determined, the ⟨text⟩ will be encapsulated with \foreignlanguage otherwise just ⟨text⟩ is done.

If \foreignlanguage isn’t defined (that is, there’s no language support for the document), this command simply does ⟨text⟩. If an old version of tracklang is used, this command issues a warning and just does ⟨text⟩.

If tracklang can’t determine the corresponding language label to use with \foreign-
5. Referencing (Using) Entries

language, then a warning is issued with:

\GlsXtrUnknownDialectWarning{⟨locale⟩}{⟨root language⟩}

where ⟨locale⟩ is the language tag supplied in the given field value and ⟨root language⟩ is the root language that tracklang has inferred from the tag.

\GlsXtrForeignText requires tracklang v1.3.6+.

For example:

\usepackage[main=british,brazilian,ngerman]{babel}
\usepackage{glossaries-extra}
\setabbreviationstyle{long-short-user}
\newabbreviation
[ user1={Associação Brasileria de Normas Técnicas},
  user2={pt-BR} ]
{abnt}{ABNT}{Brazilian National Standards Organization}
\newabbreviation
[ user1={Deutsches Institut für Normung e.V.},
  user2={de-DE-1996} ]
{din}{DIN}{German Institute for Standardization}
\newabbreviation{tug}{TUG}{\TeX\ User Group}
\renewcommand*{\glsxtruserparen}[2]{% 
  \glsxtrfullsep[#2]\
  \glsxtrparen
  {#1% 
   \ifglshasfield{\glsxtruserfield}{#2}%
   {, \ emph{\GlsXtrForeignText{#2}{\glscurrentfieldvalue}}}%  
   %
  }%}
\begin{document}
\gls{abnt}, \gls{din}, \gls{tug}.
\printunsrtglossaries
\end{document}
Example 118: Foreign language field encapsulation

Brazilian National Standards Organization (ABNT, Associação Brasileira de Normas Técnicas), German Institute for Standardization (DIN, Deutsches Institut für Normung e.V.), T\TeX User Group (TUG).

Glossary

**ABNT**  Brazilian National Standards Organization

**DIN**  German Institute for Standardization

**TUG**  T\TeX User Group

### 5.12.2. Associated Entry Format

An entry may have a particular formatting style associated with it (rather than a more general category-wide format). This needs to be provided by a text-block command that takes a single argument. The name (without the leading backslash) should be stored in the field identified by:

\begin{verbatim}
\GlsXtrFmtField
\end{verbatim}

This command should expand to the internal field label used to store the formatting command’s control sequence name. The default value is \texttt{useri} (which corresponds to the \texttt{user1} key).

\begin{verbatim}
\glsxtrfmt[⟨options⟩]{⟨entry-label⟩}{⟨text⟩}
\end{verbatim}

This command behaves like:

\begin{verbatim}
\glslink[⟨options⟩]{⟨entry-label⟩}{⟨fmt-link-text⟩}
\end{verbatim}
where the link text \textit{fmt-link-text} is formatted according to:

\glsxtrfmtdisplay{\langle csname\rangle}{\langle text\rangle}{\langle insert\rangle}

The default definition simply does \textit{\langle csname\rangle\langle text\rangle\langle insert\rangle} where the control sequence name \textit{\langle csname\rangle} is obtained from the field given by \GlsXtrFmtField. If the field hasn’t been set, \@firstofone is used (which simply does its argument). The unstarrred version assumes an empty \textit{\langle insert\rangle}. The default \glslink options are given by \GlsXtrFmtDefaultOptions.

The post-link hook is suppressed with \glsxtrfmt.

If you don’t want the complexity of \glslink, a partially expandable command is provided that may be used in section headings:

\glsxtrantryfmt{\langle entry-label\rangle}{\langle text\rangle}

If hyperref has been loaded, this will expand to:

\glsxtrpdfentryfmt{\langle entry-label\rangle}{\langle text\rangle}

within the PDF bookmarks, which just does \textit{\langle text\rangle}. Otherwise \glsxtrantryfmt will format \textit{\langle text\rangle} according to the control sequence name identified in the field given by \GlsXtrFmtField (or \@firstofone, if not set). For example:

\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage{amsmath}
\usepackage{colorlinks}\hyperref
\usepackage[postdot,style=index]{glossaries-extra}
\makeglossaries
\newcommand*{\mtx}[1]{\boldsymbol{#1}}
\newcommand*{\matxinv}{\mtx{#1}\sp{-1}}

\newglossaryentry{matrix}{{%
  name={matrix},
  symbol={\ensuremath{\mtx{M}}},
  plural={matrices},
  user1={\mtx},% corresponds to \mtx
}
5. Referencing (Using) Entries

A \gls{matrix} is denoted \glssymbol{matrix}. The inverse is denoted \glssymbol{matrixinv}.

\[
\text{A} \ \text{\gls{matrix}} \ \text{is denoted} \ \text{\glssymbol{matrix}.}
\]

Compare $\text{A}[0]$ with $\text{A}[0]$.

\begin{document}
\printglossaries
\end{document}

This produces:
Example 119: Storing a formatting command in a field

A matrix is denoted $\mathbf{M}$. The inverse is denoted $\mathbf{M}^{-1}$.

\[
\mathbf{A}\mathbf{A}^{-1} = \mathbf{I}
\]

Compare $\mathbf{A}[0]$ with $\mathbf{A}_0$.

Glossary

**identity matrix** ($\mathbf{I}$) a diagonal matrix with all diagonal elements equal to 1 and all other elements equal to 0.

**matrix** ($\mathbf{M}$) rectangular array of values.

**matrix inverse** ($\mathbf{M}^{-1}$) a square matrix such that $\mathbf{MM}^{-1} = \mathbf{I}$.

Note the difference between using \glsxtrfmt* vs \glsxtrfmt.

There are also sentence case versions of the above commands:

\begin{verbatim}
\Glsxtrfmt{(options)}{(entry-label)}{(text)}
\end{verbatim}

This is simply a shortcut for:

\begin{verbatim}
\glsxtrfmt{(options)}{(entry-label)}{\glssentencecase{(text)}}
\end{verbatim}

Similarly for the starred form:

\begin{verbatim}
\Glsxtrfmt*[{options}]{(entry-label)}{(text)}[{insert}]
\end{verbatim}

which is a shortcut for:

\begin{verbatim}
\glsxtrfmt*[{options}]{(entry-label)}{\glssentencecase{(text)}}[{insert}]
\end{verbatim}

\begin{verbatim}
\Glsxtrentryfmt{(entry-label)}{(text)}
\end{verbatim}

This is a shortcut for
5. Referencing (Using) Entries

\glsxtrentryfmt{⟨entry-label⟩}{\glssentencecase{⟨text⟩}}

but uses:

\Glsxtrpdfentryfmt{⟨entry-label⟩}{⟨text⟩}

for the PDF bookmarks. This uses \MFUsentencecase to perform the case-change, which is expandable.

If you are writing \glsxtrfmt or \glsxtrentryfmt explicitly in the document text, you can, of course, enter the appropriate case in ⟨text⟩ directly. The purpose of providing the sentence case commands is to enable a mapping to be setup with \MFUaddmap in the event that \glsxtrfmt or \glsxtrentryfmt occur at the start of content, such as another entry’s description, that will have sentence case automatically applied. This will require mfirstuc v2.08+ to support the mapping. See the mfirstuc manual for further details.

5.13. Comma-Separated Lists

These commands are for field values that are comma-separated lists (for example, the field has been constructed with \glsxtrapptocsvfield) or for testing if field values are contained within comma-separated lists.

\glsseelist{⟨csv-list⟩}

This is provided by the base glossaries package to format the entry labels in see cross-reference list. (It's used internally by \glsseeformat, which adds the see prefix.) It may also be used for any comma-separated list of entry labels. Note that the argument isn’t expanded. If expansion is required, use:

\glsxtrseelist{⟨csv-list⟩}

This fully expands its argument and passes the result to \glsseelist. With just the base glossaries package, each item is encapsulated with \glsseeitem. The glossaries-extra package redefines \glsseeitem to make it more flexible and provides additional commands to
5. Referencing (Using) Entries

further customize the formatting.

\glsxtrtaggedlist\{\langle\textit{singular tag}\rangle\}\{\langle\textit{plural tag}\rangle\}\{\langle\textit{label prefix}\rangle\}\{\langle\textit{csv-list}\rangle\}

This is a similar command that has an initial tag inserted before the start of the list. If the list only contains one element, the \langle\textit{singular tag}\rangle is used. If the list contains more than one element, the \langle\textit{plural tag}\rangle is used. The separator between the tag and the list is given by:

\glsxtrtaggedlistsep\quad initial: \textit{space}

The separators between the elements of the list and the formatting of each list element is as for \texttt{\glsseelist} (see below). If the list is empty, nothing is displayed. The \langle\textit{label prefix}\rangle is inserted before the current item in the list to form the entry label.

Spaces in \langle\textit{csv-list}\rangle are significant. Avoid unwanted leading or trailing spaces and empty labels.

\glsseeitemformat\{\langle\textit{entry-label}\rangle\}

The base glossaries package just uses \texttt{\glsentryname} or \texttt{\glsentrytext} in this command. The glossaries-extra package redefines this so that it does:

\ifglshasshort\{\langle\textit{entry-label}\rangle\}\{\texttt{\glsfmttext}\{\langle\textit{entry-label}\rangle\}\{\texttt{\glsfmtname}\{\langle\textit{entry-label}\rangle\}\}

Note that the use of \texttt{\glsfmttext} rather than \texttt{\glsentrytext} allows the abbreviation style to be used.

With glossaries-extra, the first item in \texttt{\glsseelist} will be encapsulated with:

\glsseefirstitem\{\langle\textit{entry-label}\rangle\}

The default definition is simply \texttt{\glsseeitem\{\langle\textit{entry-label}\rangle\}} but can be redefined, for example to convert the first character to uppercase if sentence case is required.
5. Referencing (Using) Entries

If the label corresponds to a multi-entry, \texttt{\textbackslash glsseefirstitem} will be used instead (see §7.12). Similarly, \texttt{\textbackslash glsseeitem} will be used instead of \texttt{\textbackslash glsseeitem} for a multi-entry label.

\textbf{\texttt{\textbackslash glsseeinitial}}

This is used between each entry in the list, except between the final pair.

\textbf{\texttt{\textbackslash glsseeelastsep}}

This is used between the penultimate and final item in the list. The default definition is:

\texttt{\textbackslash space\textbackslash andname\textbackslash space}

With glossaries-extra, if there are at least three elements in the list, the separator between the final two elements will be given by:

\textbf{\texttt{\textbackslash glsseeelastoxfordsep}}

This just defaults to \texttt{\textbackslash glsseeelastsep} but may be redefined to include a comma, if preferred.

\textbf{\texttt{\textbackslash glsxtrforcsvfield\{\langle entry-label\rangle\}\{\langle field-label\rangle\}\{\langle handler cs\rangle\}}} \quad \textbf{modifier: \texttt{\*}}

This iterates over the comma-separated list stored in the given field (identified by its internal label) for the entry identified by \langle entry-label \rangle and performs \langle handler cs\rangle\{\langle element\rangle\} for each element of the list. This command uses \texttt{\textbackslash glsxtrifhasfield} so the complete list can be obtained with \texttt{\textbackslash glscurrentfieldvalue}. Does nothing if the field hasn’t been set or the entry hasn’t been defined. The unstarred version adds implicit grouping. The starred version doesn’t.

It’s possible to prematurely break the loop at the end of the current iteration with:

\textbf{\texttt{\textbackslash glsxxtrendfor}}

If nested within another command that also uses \texttt{\textbackslash @for}, use the unstarred version to localise the break. This command is simply set to \texttt{\textbackslash @endfortrue}, which is provided by the \texttt{xfor}
5. Referencing (Using) Entries

\glsxtrfieldformatcsvlist\{entry-label\}\{field-label\}

This formats the comma-separated list stored in the given field (identified by its internal label) for the entry identified by ⟨entry-label⟩ using datatool-base’s \DTLformatlist. This command uses \glsxtrifhasfield so the complete list can be obtained with \glscurrentfieldvalue. This adds implicit grouping. There is no starred version.

The following demonstrates the difference between \glsseelist (which specifically requires a list of labels) and \glsxtrfieldformatcsvlist (which formats an arbitrary list):

\usepackage[colorlinks]{hyperref}
\usepackage[autoseeindex=false]{glossaries-extra}
\newglossaryentry{example}{name={example},description={},see={another1,another2}}
\newglossaryentry{another1}{name={another one},description={}}
\newglossaryentry{another2}{name={another two},description={}}
\begin{document}
\glsxtrapptocsvfield{example}{animals}{duck}
\glsxtrapptocsvfield{example}{animals}{albatross}
\glsxtrapptocsvfield{example}{animals}{arara}
Animal list: \glsxtrfieldformatcsvlist\{example\}\{animals\}

See list: \glsxtrifhasfield\{see\}\{example\}
{\glsxtrseelist\{\glscurrentfieldvalue\}\{not set\}.

\printunsrtglossaries
\end{document}

There’s no indexing in this document so I’ve used autoseeindex=false to avoid an error. This means there’s no cross-reference list in the glossary but, as demonstrated, the “see” list can be reproduced in the document. The result is:
Example 120: Formatting lists contained in field values

Animal list: duck, albatross & arara.
See list: another one & another two.

Glossary

eample
another one
another two

This first constructs a comma-separated list in a custom internal field with the label animals. There’s no associated key that can be used in \newglossaryentry. In this case, the field could simply be set in one command. For example:

\glsxtrdeffield{example}{animals}{duck,albatross,arara}

The main reason for providing \glsxtrapptocsvfield is for the benefit of bib2gls, as it sometimes has to construct a field value list while it’s writing the glstex file, but there may be other uses in complex documents that construct field values through some custom function.

\GlsXtrIfValueInFieldCsvList{⟨entry-label⟩}{⟨field-label⟩}{⟨value⟩}{⟨true⟩}{⟨false⟩} modifier: *

This does ⟨true⟩ if the comma-separated list stored in the given field (identified by its internal label) contains the given ⟨value⟩ (using \DTLifinlist provided by datatool-base) or ⟨false⟩ if the value isn’t in the list or if the field hasn’t been set or the entry hasn’t been defined. The unstarred version adds implicit grouping. The starred version doesn’t.

This command internally uses \glsxtrifhasfield, so take care if it’s nested. Within ⟨false⟩, you can test if \glscurrentfieldvalue is empty or undefined. If it’s defined but not empty, then the field has been set but doesn’t contain ⟨value⟩.

\GlsXtrIfFieldValueInCsvList{⟨entry-label⟩}{⟨field-label⟩}{⟨csv-list⟩}{⟨true⟩}{⟨false⟩} modifier: *

This command is essentially the other way around to the above. In this case, the comma-
separated list is provided in the argument \langle csv-list \rangle and the search value is the field’s value. This does \langle true \rangle if the value is found in \langle csv-list \rangle or \langle false \rangle if the value isn’t in \langle csv-list \rangle or the field isn’t set or the entry hasn’t been defined. The unstarred version adds implicit grouping. The starred version doesn’t. Again, this command internally uses \texttt{\textbackslash glsXtrHasField}, so you can test \texttt{\textbackslash glsCurrentFieldValue} in \langle false \rangle to determine whether or not the field has been set.

\begin{verbatim}
\xGlsXtrIfValueInFieldCsvList{\langle entry-label \rangle}{\langle field-label \rangle}{\langle value \rangle}{\langle true \rangle}{\langle false \rangle}
\end{verbatim}

As \texttt{\textbackslash GlxStrIfValueInFieldCsvList} but fully expands \langle value \rangle first.

### 5.14. List Fields

Comma-separated list fields are covered in §5.13. The commands in this section are for fields that store etoolbox internal lists. Elements can be appended to these fields using commands \texttt{\textbackslash glsXtrFieldListAdd}, described in §3.5. The commands listed below provide an easy interface to iterate over the field values. See the etoolbox documentation for further details about internal lists.

\begin{verbatim}
\glsxtrFieldFormatList{\langle entry-label \rangle}{\langle field-label \rangle}
\end{verbatim}

Formats the list using the same separators as used by datatool’s \texttt{\textbackslash DTLFormatList}. This internally uses etoolbox’s \texttt{\textbackslash forListCSLoop} with the same handler macro as used with \texttt{\textbackslash DTLFormatList}.

\begin{verbatim}
\glsxtrFieldDolistLoop{\langle entry-label \rangle}{\langle field \rangle}
\end{verbatim}

This uses etoolbox’s \texttt{\textbackslash doListCSLoop}, which uses the command \texttt{\textbackslash do} as the handler.

\begin{verbatim}
\glsxtrFieldForListLoop{\langle entry-label \rangle}{\langle field \rangle}{\langle handler-cs \rangle}
\end{verbatim}

This uses etoolbox’s \texttt{\textbackslash forListCSLoop}, which uses the \langle handler-cs \rangle as the handler.

\begin{verbatim}
\glsxtrFieldIfInList{\langle entry-label \rangle}{\langle field \rangle}{\langle item \rangle}{\langle true \rangle}{\langle false \rangle}
\end{verbatim}
This uses etoolbox’s \ifinlistcs to test if ⟨item⟩ is in the list.

\glsxtrfieldxifinlist{⟨entry-label⟩}{⟨field⟩}{⟨item⟩}{⟨true⟩}{⟨false⟩}

This uses etoolbox’s \ifinlistcs to test if ⟨item⟩ is in the list.

5.15. Field Conditionals

\GlsXtrIfFieldUndef{⟨field-label⟩}{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}

Tests if the given field (identified by its internal label) is undefined for the entry given by ⟨entry-label⟩. Does ⟨true⟩ if the entry doesn’t exist or if entry exists but the field hasn’t been set. Does ⟨false⟩ if the field has been set, even if it has been set to empty. Unlike \glesxtrifhasfield there is no grouping or starred version and no assignment of \glscurrentfieldvalue. This is simply a shortcut that internally uses etoolbox’s \ifcsundef. The base glossaries package provides a similar command \ifglsfieldvoid, which uses etoolbox’s \ifcsvoid instead.

\glesxtrifhasfield{⟨field-label⟩}{⟨entry-label⟩}{⟨true⟩}{⟨false⟩} modifier: *

This tests if the entry given by ⟨entry-label⟩ has the field identified by its internal label ⟨field-label⟩ set. This is like \ifglshasfield but doesn’t produce a warning if the entry or field doesn’t exist.

This command first assigns \glscurrentfieldvalue to the field value. If this is defined and not empty, ⟨true⟩ is done otherwise ⟨false⟩ is done. You can test \glscurrentfieldvalue within ⟨false⟩ to find out whether it’s undefined or empty using etoolbox’s commands, such as \ifdefempty or \ifundef.

The unstarred version adds implicit grouping to make nesting easier. The starred version doesn’t (to make assignments easier).

If you are simply displaying the value of the field (for example, in the post-description hook) then use the unstarred version. If you are making an assignment based on the value of the field, then use the starred version.

\GlsXtrIfFieldCmpNum{⟨field-label⟩}{⟨entry-label⟩}{⟨op⟩}{⟨number⟩}{⟨true⟩}{⟨false⟩} modifier: *

305
This command should only be used with fields that contain integer values. It internally uses \glsxtrifhasfield (the starred or unstarred version, to match the starred or unstarred version of \GlsXtrIfFieldEqStr) and tests if \glscurrentfieldvalue is equal to \langle op \rangle is =, less than \langle op \rangle is < or greater than \langle op \rangle is > the given number \langle number \rangle. If the field is empty or undefined, \glscurrentfieldvalue will be set to 0. Remember that the unstarred version adds implicit grouping.

\GlsXtrIfFieldEqNum\{field-label\}\{entry-label\}\{number\}\{true\}\{false\}

This is a shortcut that uses \GlsXtrIfFieldCmpNum with \langle op \rangle set to =. The unstarred version adds implicit grouping.

\GlsXtrIfFieldNonZero\{field-label\}\{entry-label\}\{true\}\{false\}

This is a shortcut that uses \GlsXtrIfFieldCmpNum with \langle op \rangle set to = and the final two arguments swapped. (So it’s true if the field value is not zero.) The unstarred version adds implicit grouping.

\GlsXtrIfFieldEqStr\{field-label\}\{entry-label\}\{value\}\{true\}\{false\}

This internally uses \glsxtrifhasfield (the starred or unstarred version, to match the starred or unstarred version of \GlsXtrIfFieldEqStr) and tests if \glscurrentfieldvalue is equal to \langle value \rangle. Remember that the unstarred version adds implicit grouping.

\GlsXtrIfFieldEqXpStr\{field-label\}\{entry-label\}\{value\}\{true\}\{false\}

This is like \GlsXtrIfFieldEqStr but expands the string before the comparison. This also has an starred version that doesn’t add implicit grouping.

\GlsXtrIfXpFieldEqXpStr\{field-label\}\{entry-label\}\{value\}\{true\}\{false\}

This is like \GlsXtrIfFieldEqStr but expands both the field value and the string before the comparison. This also has an starred version that doesn’t add implicit grouping.
6. Counting References

There are three basic ways of counting entry references:

1. Counting the total number of times \glsunset is used (\glsreset resets the count unless \glsresetcurrcountfalse and is best avoided). This is provided by the base glossaries package and is intended for documents where the term should be displayed differently if it’s only been used a certain number of times. The information has to be written to the aux file so that it’s available on the next \LaTeX run.

   This method is extended by glossaries-extra and is described in §6.1. This method relies on the document only using the \gls-like commands and is inappropriate with \bib2gls.

2. Counting the total number of records. This method is only available with \bib2gls and is intended for documents where the term should be displayed differently if it’s only been recorded (indexed) a certain number of times. This is a more efficient method than entry counting. See §11.4 for further details.

3. Counting the number of times the \gls-like or \glstext-like commands are used. Unlike the other two methods, this just provides a running total rather than the total from the previous \LaTeX run. This method is intended to make it more convenient to work with hooks like \glslinkcheckfirsthyperhook, \glslinkpostsetkeys or \glslinkpresetkeys. See §6.2 for further details.

6.1. Entry Counting (First Use Flag)

   If you are using \bib2gls, you need to use record counting instead (see §11.4).

   Entry counting is provided by the base glossaries package and is enabled with \glsenableentrycount. This keeps a count of the number of times an entry is marked as used, which is done by hooking into the unset and reset commands (see §5.10). The current running total can be obtained with \glsentrycurrcount. The total from the end of the previous \LaTeX run can be obtained with \glsentryprevcount.

   Since entry counting relies on the first use flag, it doesn’t take the \glstext-like commands into account.
6. Counting References

Entry counting is incompatible with \texttt{docdef=true}.

The \texttt{glossaries-extra} package modifies \texttt{\glsenableentrycount} to allow for the \texttt{entry-count} attribute. This means that you not only need to enable entry counting with \texttt{\glsenableentrycount}, but you also need to set the \texttt{entrycount} attribute (see below).

Prior to v1.49, the associated counter was reset back to 0 when the first use flag is reset. This behaviour is now only implemented if the following conditional is true:

\begin{verbatim}
\ifglsresetcurrcount \langle true\rangle \else \langle false\rangle \fi initial: \iffalse
\end{verbatim}

To (locally) change this conditional to true use:

\begin{verbatim}
\glsresetcurrcounttrue
\end{verbatim}

To (locally) change this conditional to false use:

\begin{verbatim}
\glsresetcurrcountfalse
\end{verbatim}

As from v1.49, the default is now false. Note that this conditional is also available with \texttt{glossaries v4.50+}.

Remember that entry counting only counts the number of times an entry is used by commands that change the first use flag. (That is, all those commands that mark the entry as having been used.) There are many commands that don’t modify this flag and they won’t contribute to the entry use count.

With just the base \texttt{glossaries} package, the associated entry counting commands, such as \texttt{\cgls}, are only available when entry counting has been activated with \texttt{\glsenableentrycount}. Whereas with \texttt{glossaries-extra}, those commands are always available but behave in the same way as the corresponding \texttt{\gls}-like commands if entry counting hasn’t been activated. The commands provided by the \texttt{shortcuts} options, such as \texttt{\ac} are defined to use \texttt{\cgls} instead of \texttt{\gls} etc so you can use them either with or without entry counting.

In order to activate entry counting with \texttt{glossaries-extra}, you not only need to use \texttt{\glsenableentrycount} but also need to specify the trigger value.

\begin{verbatim}
\GlsXtrEnableEntryCounting\langle category-list\rangle\{\langle trigger-value\rangle\}
\end{verbatim}
6. Counting References

This command is provided as a shortcut to activate entry counting and assign the trigger value. This command performs the following:

- enables entry counting with \glxtrenableentrycount;
- redefines the \gls-like commands to do the equivalent \cgls commands (so you don’t need to keep track of which entries have entry counting enabled);
- sets the entrycount attribute to \langle trigger-value \rangle for all the supplied categories;
- disables the unit counting command (which is incompatible).

If you want to have different trigger values for different categories, you can set the entrycount attribute afterwards for the other category. For example:

\begin{verbatim}
\GlsXtrEnableEntryCounting{abbreviation,acronym}{1}
\glssetcategoryattribute{general}{2}
\end{verbatim}

If you use \GlsXtrEnableEntryCounting multiple times, the repeated instances will simply set the entrycount attribute for the listed categories. So the above can also be written as:

\begin{verbatim}
\GlsXtrEnableEntryCounting{abbreviation,acronym}{1}
\GlsXtrEnableEntryCounting{general}{2}
\end{verbatim}

The commands like \cgls behave like the corresponding \gls-like command if the entry count at the end of the previous run was more than a trigger value. With just the base glossaries package, this trigger value is 1. With glossaries-extra you can specify a different value.

The appropriate trigger value must be set for the required category or categories.

As with the \gls-like commands, the \cgls set of commands may also be used with the star (\*) or plus (+) modifiers or the modifier given by \GlsXtrSetAltModifier.

If the entry count at the end of the previous run doesn’t exceed the trigger value, the corresponding formatting command is used instead. For example, \cgls will use \cgls-format. The complete set of commands are:

\begin{verbatim}
\cgls[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle] modifiers: * + \langle alt-mod \rangle
\end{verbatim}
6. Counting References

If the trigger value has been supplied for the entry’s category and is exceeded, this behaves like \texttt{\gls} otherwise it uses:

\begin{verbatim}
\glsformat{⟨entry-label⟩}{⟨insert⟩}
\end{verbatim}

This is redefined by glossaries-extra to test whether or not the entry has the regular attribute set or is an abbreviation:

\begin{verbatim}
\renewcommand*{\glsformat}[2]{%\glsifregular{#1}{\glsentryfirst{#1}}%\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%2%}
\end{verbatim}

This show the first use value if the entry is regular otherwise it will show the long form. The insert is appended at the end.

\begin{verbatim}
\glspl{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} \textbf{modifiers:} * + ⟨alt-mod⟩
\end{verbatim}

If the trigger value has been supplied for the entry’s category and is exceeded, this behaves like \texttt{\glspl} otherwise it uses:

\begin{verbatim}
\glsplformat{⟨entry-label⟩}{⟨insert⟩}
\end{verbatim}

This is like \texttt{\glsformat} but uses the plural commands.

\begin{verbatim}
\Gls{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} \textbf{modifiers:} * + ⟨alt-mod⟩
\end{verbatim}

If the trigger value has been supplied for the entry’s category and is exceeded, this behaves like \texttt{\Gls} otherwise it uses:

\begin{verbatim}
\Glsformat{⟨entry-label⟩}{⟨insert⟩}
\end{verbatim}

This is like \texttt{\glsformat} but uses the sentence case commands.

\begin{verbatim}
\Glspl{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} \textbf{modifiers:} * + ⟨alt-mod⟩
\end{verbatim}

If the trigger value has been supplied for the entry’s category and is exceeded, this behaves
6. Counting References

like \Glspl otherwise it uses:

\cGlsplformat{⟨entry-label⟩}{⟨insert⟩}

This is like \cglsformat but uses the plural sentence case commands.

The glossaries-extra package provides some additional commands:

\cGLS[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

If the trigger value has been supplied for the entry’s category and is exceeded, this behaves like \GLS otherwise it uses:

\cGLSformat{⟨entry-label⟩}{⟨insert⟩}

This simply uses \cglsformat converted to uppercase.

\cGLSpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩

If the trigger value has been supplied for the entry’s category and is exceeded, this behaves like \GLSpl otherwise it uses:

\cGLSplformat{⟨entry-label⟩}{⟨insert⟩}

This simply uses \cglsplformat converted to uppercase.

The test to determine whether or not an entry trips the trigger value is performed by:

\glsxtrifcounttrigger{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}

This obtains the trigger value from the entry’s entrycount attribute.

Since these commands require information from the previous \LaTeX run, and extra \LaTeX call must be added to the build process (before the relevant indexing application).

For example, in the following document the trigger value is set to 1. The CSS entry is only used once (which doesn’t exceed the trigger). The HTML entry is used twice (which does exceed the trigger). The sample entry is only used once, but entry counting hasn’t been enabled on its category (the default general).
6. Counting References

\usepackage{colorlinks}{hyperref}
\usepackage{glossaries-extra}
\makeglossaries
\GlsXtrEnableEntryCounting{abbreviation}{1}
\newabbreviation{css}{CSS}{cascading style sheet}
\newabbreviation{html}{HTML}{hypertext markup language}
\newglossaryentry{sample}{name={sample},description={an example}}
\begin{document}
First use: \gls{css}, \gls{html} and \gls{sample}.
Next use: \gls{html}.
\printglossaries
\end{document}

If the document is saved in a file called myDoc.tex then the build process is:

\begin{verbatim}
pdflatex myDoc
pdflatex myDoc
makeglossaries myDoc
pdflatex myDoc
\end{verbatim}

Note the second \LaTeX call before makeglossaries. The result is shown below:

Example 121: Entry counting according to category

First use: cascading style sheet, hypertext markup language (HTML) and sample. Next use: HTML.

Glossary

HTML hypertext markup language 1
sample an example 1

Note that the CSS entry only shows the long form, doesn’t appear in the glossary and doesn’t have a hyperlink. This is because the total count from the previous \LaTeX run doesn’t exceed the value (1, in this case) that triggers the normal behaviour of \gls. The HTML entry has a total count of 2 from the previous \LaTeX run, so it’s displayed as normal with the full form on first use and has a hyperlink to its entry in the glossary.

The sample entry is only used once, but it has the default general category, which doesn’t have the entrycount attribute set.
Note that if the build process only had one \LaTeX call before running \texttt{makeglossaries}, the HTML entry would also not appear in the glossary. This is because on the first \LaTeX run, the total from the previous run is 0 (because there’s no information in the aux file).

The \texttt{glossaries-extra} package also provides the ability to count per sectional unit instead:

\begin{verbatim}
\glsenableentryunitcount
\end{verbatim}

It’s not possible to enable both document-wide entry counting (\texttt{\glsenableentrycount}) and unit entry counting (\texttt{\glsenableentryunitcount}).

The unit entry counting provides separate totals for each section unit. As above, this uses the \texttt{entrycount} attribute to provide the trigger value but also requires the \texttt{unitcount} attribute, which should be set to the name of the appropriate counter, such as section or chapter.

Due to the asynchronous nature of \TeX’s output routine, discrepancies will occur in page spanning paragraphs if you use the page counter.

As before, there is a command provided to enable the feature and set the corresponding attributes at the same time:

\begin{verbatim}
\GlsXtrEnableEntryUnitCounting{\langle category-list} \langle trigger-value} \langle counter}\end{verbatim}

This command performs the following:

- enables unit entry counting with \texttt{\glsenableentryunitcount};
- redefines the \texttt{\gls} like commands to do the equivalent \texttt{\cgls} commands (so you don’t need to keep track of which entries have entry counting enabled);
- sets the \texttt{entrycount} attribute to the supplied trigger for all the supplied categories;
- sets the \texttt{unitcount} attribute to the supplied counter for all the supplied categories;
- disables the document-wide counting command (which is incompatible).

If you use \texttt{\GlsXtrEnableEntryUnitCounting} multiple times, the repeated instances will simply set the \texttt{entrycount} and \texttt{unitcount} attributes for the listed categories.

The counter value is used as part of a label, which means that \texttt{\the\langle counter-name\rangle} needs to be expandable. Since \texttt{hyperref} also has a similar requirement and provides \texttt{\the\langle counter-name\rangle} as an expandable alternative, \texttt{glossaries-extra} will use \texttt{\the\langle counter-name\rangle} if it exists otherwise it will use \texttt{\the\langle counter-name\rangle}.
6. Counting References

The commands for accessing the totals, `\glsentrycurrcount` and `\glsentryprevcount` have different definitions with unit entry counting and will expand to the total for the current unit. The overall totals can be obtained with additional commands:

\[ \text{\glstextentrycount{\{entry-label\}}} \]

This expands to the overall total from the previous \LaTeX{} run.

\[ \text{\glstextentrymaxcount{\{entry-label\}}} \]

This expands to the maximum per-unit total from the previous \LaTeX{} run.

For example:

\begin{verbatim}
\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\GlsXtrEnableEntryUnitCounting{abbreviation}{2}{section}
\makeglossaries
% category={abbreviation}:
\newabbreviation{html}{HTML}{hypertext markup language}
\newabbreviation{css}{CSS}{cascading style sheet}
% category={general}:
\newglossaryentry{sample}{name={sample},description={sample}}
\begin{document}
\section{Sample}
Used once: \gls{html}.
Used three times: \gls{css} and \gls{css} and \gls{css}.
Used once: \gls{sample}.
\section{Another Sample}
Used once: \gls{css}.
\end{document}
\end{verbatim}
6. Counting References

\begin{flushleft}
Used twice: \texttt{\gls{html}} and \texttt{\gls{html}}.
\end{flushleft}
\printglossaries
\end{document}

As before, if the document is in a file called \texttt{myDoc.tex} then the build process is:

\begin{flushleft}
\texttt{pdflatex myDoc}
\texttt{pdflatex myDoc}
\texttt{makeglossaries myDoc}
\texttt{pdflatex myDoc}
\end{flushleft}

The result is:

\begin{flushleft}
Example 122: Entry unit counting (per section) according to category

\section{Sample}

Used once: hypertext markup language.
  Used three times: cascading style sheet (CSS) and CSS and CSS.
  Used once: sample.

\section{Another Sample}

Used once: cascading style sheet.
  Used twice: hypertext markup language and hypertext markup language.

\textbf{Glossary}

\begin{flushleft}
\textbf{CSS} cascading style sheet 1
\textbf{sample} an example 1
\end{flushleft}
\end{flushleft}

In this document, the CSS entry is used three times in the first section. This is more than the trigger value of 2, so \texttt{\gls{css}} is expanded on first use with the short form used on subsequent use, and the CSS entries in that section are added to the glossary. In the second section, the CSS entry is only used once, which trips the suppression trigger, so in that section, the long form is used and \texttt{\gls{css}} doesn’t get a line added to the glossary file.

The HTML entry is used a total of three times, but the expansion and indexing suppression trigger is tripped in both sections because the per-unit total (1 for the first section and 2 for the second chapter) is less than or equal to the trigger value.
6. Counting References

The sample entry has only been used once, but it doesn’t trip the indexing suppression because it’s in the \textit{general} category, which hasn’t been listed in \texttt{\GlsXtrEnableEntryUnitCounting}.

The per-unit entry counting can be used for other purposes. In the following example document, the trigger value is set to zero, which means the index suppression won’t be triggered, but the unit entry count is used to automatically suppress the hyperlink for commands like \texttt{\gls} by modifying the \texttt{\glslinkcheckfirsthyperhook} which is used at the end of the macro the determines whether or not to suppress the hyperlink.

\begin{verbatim}
\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\makeglossaries
\GlsXtrEnableEntryUnitCounting{general}{0}{page}
\newglossaryentry{sample}{name={sample},description={an example}}
\renewcommand*{\glslinkcheckfirsthyperhook}{\ifnum\glsentrycurrcount\glslabel>0\setupglslink{hyper=false}\fi}
\begin{document}
A \gls{sample} entry. Next use: \gls{sample}.

\newpage
Next page: \gls{sample}. Again: \gls{sample}.
\end{document}
\end{verbatim}

This only produces a hyperlink for the first instance of \texttt{\gls{sample}} on each page.
Example 123: Enabling unit counting to hook into hyperlink setting

A sample entry. Next use: sample.

Glossary

sample an example 1, 2

The earlier warning about using the page counter still applies. If the first instance of \texttt{\gls} occurs at the top of the page within a paragraph that started on the previous page, then the count will continue from the previous page.

### 6.2. Link Counting

As from version 1.26, an alternative method of entry counting is to count the number of times the \texttt{\gls}-like or \texttt{\glstext}-like commands are used. (The "link" in this method’s name refers to the use of the internal command \texttt{@gls@link} not to \texttt{\hyperlink} although \texttt{@gls@link} may use \texttt{\hyperlink} when displaying the link text.)

To enable link counting use the preamble-only command:

\begin{verbatim}
\GlsXtrEnableLinkCounting[\texttt{(parent counter)}]{\texttt{(categories)}}
\end{verbatim}

where \texttt{(categories)} is a list of category labels. The optional argument \texttt{(parent counter)} may be used to identify a parent counter (which must already be defined). If present, the associated link counter will be reset when the parent counter is incremented. This command automatically sets the \texttt{linkcount} attribute for the given categories. If the optional argument is present, it also sets the \texttt{linkcountmaster} attribute.

When enabled, the \texttt{\gls}-like and \texttt{\glstext}-like commands will increment the associated counter using

\begin{verbatim}
\glsxtrinclinkcounter{\texttt{(counter)}}
\end{verbatim}

This just does \texttt{\stepcounter{\texttt{(counter)}}} by default but if you need \texttt{\refstepcounter} instead, just redefine this command:

\begin{verbatim}
\renewcommand*{\glsxtrinclinkcounter}[1]{\refstepcounter{#1}}
\end{verbatim}
You can access the internal count register using:

\GlsXtrLinkCounterValue{⟨entry-label⟩}

where ⟨label⟩ is the entry’s label. This will expand to 0 if the register hasn’t been defined.

It’s also possible to access the display value (\the⟨counter⟩) using

\GlsXtrTheLinkCounter{⟨entry-label⟩}

(This will expand to 0 if the counter hasn’t been defined.)

In order to conserve resources, the counter is only defined when it first needs to be incremented so terms that have been defined but haven’t been used in the document won’t have the associated count register allocated.

You can test if the counter has been defined using:

\GlsXtrIfLinkCounterDef{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}

This expands to ⟨true⟩ if the link counter associated with the entry identified by ⟨entry-label⟩ has been defined, otherwise expands to ⟨false⟩.

The counter name can be obtained using

\GlsXtrLinkCounterName{⟨entry-label⟩}

This simply expands to the counter name associated with the entry given by ⟨entry-label⟩ without any check for existence. For example, to change the display command (\the⟨counter⟩) using etoolbox:

\csdef{the\GlsXtrLinkCounterName{duck}}%\Roman{\GlsXtrLinkCounterName{duck}}

This is useful if you just want to change the display for specific entries but isn’t convenient if you want to change the display for all entries. Instead, it’s simpler to redefine \GlsXtrTheLinkCounter. For example:
6. Counting References

In both cases, the redefinition should be implemented after \GlsXtrEnableLinkCounting.

Here’s an example document that uses link counting to disable the hyperlink after the first reference. This redefines \glslinkpresetkeys (which is used by both \gls and \glstext) instead of \glslinkcheckfirsthyperhook (which is used by \gls but not by \glstext).

\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage{glossaries-extra}
\makeglossaries
\renewcommand*{\glslinkpresetkeys}{%\ifnum\GlsXtrLinkCounterValue{\glslabel}>1 \setupglslink{hyper=false}\fi}
\GlsXtrEnableLinkCounting{general}
\newglossaryentry{sample1}{name={sample1},description={an example}}
\newglossaryentry{sample2}{name={sample2},description={another example}}
\newabbreviation{ex}{ex}{example}
\begin{document}
\section{Sample Section}
\gls{sample1}, \gls{sample2} and \gls{ex}.
\glstext{sample1} and \gls{ex} again.
\section{Another Sample Section}

\end{document}
The use of `\gls{sample1}`, `\gls{sample2}` and `\gls{ex}`.

The use of `\glslinkpresetkeys` means that the options can override this. For example

\begin{verbatim}
\gls{hyper=true}{sample1}
\end{verbatim}

(or simply `\gls{+sample1}`) will override the `hyper=false` setting in `\glslinkpresetkeys`. If `\glslinkpostsetkeys` is used instead, the `hyper=false` setting will override the setting provided in the optional argument. The resulting document is shown below:

Example 124: Link counting used to selectively suppress hyperlinks

1 Sample Section

Sample1, sample2 and example (ex). Sample1 and ex again.

2 Another Sample Section

Sample1, sample2 and ex.

Glossary

ex example 1

sample1 an example 1

sample2 another example 1

The abbreviation category doesn’t have the `linkcount` attribute set (since it’s not listed in the argument of `\GlsXtrEnableLinkCounting`). This means that `\GlsXtrLinkCounterValue` always expands to 0 for the abbreviation (ex), so the inequality test:

\begin{verbatim}
\ifnum\GlsXtrLinkCounterValue{\glslabel}>1
\end{verbatim}

will always be false. This means that the abbreviation won’t have `hyper=false` applied. If the test is changed to
Then the abbreviation will always have `hyper=false` applied.
To reset the counter every section use the optional argument to set the parent counter:

\begin{verbatim}
\GlsXtrEnableLinkCounting[section]{general}
\end{verbatim}
7. Multi (or Compound) Entries

Nested entries (where the entry definition references other entries) are discussed in §5.4. This chapter deals with occasions where a term or phrase may consist of multiple sub-terms that are independently defined. (Examples in §7.1.5 and §7.1.6 provide workarounds for nested entries.)

For example, the names of bacteria, such as *Clostridium botulinum* and *Clostridium perfringens*, are made up of the genus (for example, *Clostridium*) and the species (for example, *botulinum* or *perfringens*). The genus is often abbreviated after first use. For example, *C. botulinum*. However, if the name is defined as a single term consisting of both the genus and species then it’s not possible to apply the abbreviation when a different species with the same genus is used. Consider the following document:

```
\documentclass{article}
\usepackage{glossaries-extra}
\setabbreviationstyle{long-only-short-only}
\newabbreviation{cbot}{C. botulinum}{Clostridium botulinum}
\newabbreviation{cperf}{C. perfringens}{Clostridium perfringens}
\begin{document}
\gls{cbot}, \gls{cbot}, \gls{cperf}.
\end{document}
```

The result is:

```
Clostridium botulinum, C. botulinum, Clostridium perfringens.
```

However, it should more typically be:

```
Clostridium botulinum, C. botulinum, C. perfringens.
```

In this case, the genus should actually be a separate definition:
7. Multi (or Compound) Entries

This is quite awkward to write. This chapter describes how to provide a shortcut for this kind of construct. Each term should be defined as normal (as in the above example), and a “multi-entry” label is then defined with the list of labels of the entries that need to be referenced.

\multiglossaryentry[⟨options⟩]{⟨multi-label⟩}{⟨main-label⟩}{⟨entry-label-list⟩}

This defines a multi-entry set with the label ⟨multi-label⟩, consisting of the entries whose labels are listed in ⟨entry-label-list⟩, where the main entry (which must be present in ⟨entry-label-list⟩) is identified by ⟨main-label⟩. If ⟨main-label⟩ is omitted, it’s assumed to be the final label in ⟨entry-label-list⟩. The main entry is described in more detail in §7.2.

The entries in ⟨entry-label-list⟩ must already be defined (using commands like \newglossaryentry or \newabbreviation).

The ⟨options⟩ are a comma-separated list of options to override the current settings and are described in §7.9.

The earlier example can now be modified to include the following:

\multiglossaryentry{cbot}{clostridium,botulinum}
\multiglossaryentry{cperf}{clostridium,perfringens}

These commands must come after the clostridium, botulinum and perfringens definitions.

Once defined, a multi-entry set can be referenced in the document using commands like:

\mglsh[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

This command essentially does \gls{⟨label⟩} for each item in the ⟨label list⟩ (with separators, see §7.4). If the final optional argument ⟨insert⟩ is provided, it will be applied to the
7. Multi (or Compound) Entries

final (non-skipped) element in the list. So the document body in the above example, can be rewritten as:

\begin{Verbatim}
\mglscbot, \mglscbot, \mglscperf.
\end{Verbatim}

There are some variants of \mglst listed in §7.11. The available \textit{options} are listed in §7.10. They are applied after the \texttt{\multiglossaryentry} options and will override settings for the individual entries.

You can't use \textit{⟨multi-label⟩} in commands like \texttt{\gls} as this label represents a set of entry labels not a single entry.

The \texttt{\multiglossaryentry} command will generate an error if the label has already been defined as a multi-entry.

\begin{Verbatim}
\provide\multiglossaryentry[⟨options⟩]{⟨multi-label⟩}{⟨main-label⟩}{⟨entry-label-list⟩}
\end{Verbatim}

This does nothing if a multi-entry set with the given label has already been defined otherwise it will act like \texttt{\multiglossaryentry}. Notes and associated commands applying to \texttt{\multiglossaryentry} also apply to \texttt{\provide\multiglossaryentry} unless otherwise stated.

\texttt{\multiglossaryentry} may be placed anywhere after the entries listed in \textit{⟨label list⟩} have been defined. A multi-entry label can’t be referenced (with commands like \texttt{\mglst}) before it has been defined.

There is limited support for \texttt{docdef=true}. The multi-entry definition can be picked up from the aux file on the next run to allow cross-references in any glossaries that occur at the start of the document. Any changes made with commands like \texttt{\mglssetmain} won’t be carried over to the next run.

By default \texttt{\multiglossaryentry} will be localised to the current scope. If you want to globally define a multi-entry you need to first switch on global definitions with:

\begin{Verbatim}
\multiglossaryentryglobaltrue
\end{Verbatim}
7. Multi (or Compound) Entries

To switch back to local definitions use:

\multiglossaryentryglobalfalse

You can test if this setting is on with:

\ifmultiglossaryentryglobal (true) \else (false) \fi

If you want to change the multi-entry options (locally) you can use:

\mglsSetOptions\{\multi-label\}\{\new-options\}

This removes the original options and replaces them with \(\text{new-options}\). If you want to (locally) append to the existing options, use:

\mglsAddOptions\{\multi-label\}\{\new-options\}

Note that \multiglossaryentry doesn’t make any adjustments to the component entries. You will need to use the parent key when you define the entries if you want a hierarchical structure in your glossary. (See the example in §7.1.1.)

If you don’t want the other elements in the glossary, you can suppress the indexing with indexothers=false (§7.9.1) or put them in an ignored glossary. For example:

\newignoredglossary\{common\}
\newabbreviation[type={common}]\{clostridium\}\{C.\}\{Clostridium\}

The \(\text{multi-label}\) can’t be used in commands like \texttt{\gls} since the label refers to a set of entry labels not to an individual entry. Similarly, an individual entry label can’t be used in commands like \texttt{\mgls}. It is possible (although potentially confusing) to use the same label for a multi-entry as for an individual entry (see the example in §7.1.6). Context will determine which is meant, except in the case of the cross-referencing fields (see, seealso and alias) where the cross-referenced label will first be tested if it’s a known multi-entry label.

If you don’t want to have to keep track of which labels refer to multi-entries and which refer to individual entries you can use:

\GlsXtrMglsOrGls\{\mgls cs\}\{\gls cs\}\{\text{modifier}\} \{\text{options}\} \{\text{label}\} \{\text{insert}\}
7. Multi (or Compound) Entries

where \textit{⟨mgls cs⟩} is the \texttt{\textbackslash mgls} like command to use if \textit{⟨label⟩} has been defined as a multi-entry and \textit{⟨gls cs⟩} is the \texttt{\textbackslash gls} like or \texttt{\textbackslash glstext} like command to use otherwise. The \textit{⟨modifier⟩} may be omitted, otherwise it’s the modifier that may be used with \texttt{\textbackslash mgls} or \texttt{\textbackslash gls} (asterisk \texttt{*}, plus \texttt{+} or the token identified with \texttt{\GlsXtrSetAltModifier}). The modifier and remaining options are passed to the relevant command (\texttt{⟨mgls cs⟩} or \texttt{⟨gls cs⟩}).

You may prefer to define your own shortcut commands for common combinations. For example, (assuming these commands haven’t already been defined by the \texttt{shortcuts} option):

\begin{verbatim}
\newcommand{\ac}{\GlsXtrMglsOrGls{\mgls}{\gls}}
\newcommand{\acp}{\GlsXtrMglsOrGls{\mglsmainpl}{\glsp}}
\newcommand{\Ac}{\GlsXtrMglsOrGls{\Mgls}{\Gls}}
\newcommand{\Acp}{\GlsXtrMglsOrGls{\Mglsmainpl}{\Glsp}}
\end{verbatim}

7.1. Examples

7.1.1. Example: Hierarchical

Bacteria names are represented by the genus (for example, Clostridium) followed by the species (for example, botulinum). This example has the genus as a parent of the species.

\begin{verbatim}
\documentclass{article}
\usepackage[titlecolorlinks]{hyperref}
\usepackage[stylemods=bookindex,style=bookindex]{glossaries-extra}
\makeglossaries
\newcommand{\latinname}[1]{\emph{#1}}
\glssetcategoriesattributes
{genus,species}{textformat,glossnamefont}{latinname}
\setabbreviationstyle[genus]{long-only-short-only-desc}
\newabbreviation
 [category={genus},description={}] {clostridium}{C.}{Clostridium}
\newglossaryentry{botulinum}{name={botulinum},category={species},description={},parent={clostridium}}
\newglossaryentry{perfringens}{name={perfringens},
\end{verbatim}
7. *Multi (or Compound) Entries*

\begin{document}
\newglossaryentry{tetani}{name={tetani}, category={species}, description={}, parent={clostridium}}
\multiglossaryentry{cbot}{clostridium,botulinum}
\multiglossaryentry{cperf}{clostridium,perfringens}
\multiglossaryentry{ctet}{clostridium,tetani}
\multiglossaryentrysetup{indexothers=false, hyper=allmain}

\begin{document}
First use: \texttt{\mgls{cbot}}, \texttt{\mgls{cperf}}, \texttt{\mgls{ctet}}.

Next use: \texttt{\mgls{cbot}}, \texttt{\mgls{cperf}}, \texttt{\mgls{ctet}}.

\printglossaries
\end{document}

This suppresses the indexing of the non-main elements (in this case, the genus). However the genus is included in the glossary (without a location list) because it’s the parent of the species (which are indexed).

Example 125: Multi-entries: hierarchical

First use: *Clostridium botulinum, C. perfringens, C. tetani.*
Next use: *C. botulinum, C. perfringens, C. tetani.*

\textbf{Glossary}

\begin{center}
\begin{tabular}{lr}
\hline
C & \textit{perfringens} 1 \\
\textit{Clostridium} & \textit{tetani} 1 \\
\textit{botulinum} & 1 \\
\hline
\end{tabular}
\end{center}

The \texttt{hyper=allmain} option makes the entire content of each \texttt{\mgls} a hyperlink to the main entry in the glossary.

7.1.2. Example: Suffix

This example has a minor modification to the previous one. In this case the multi-entries are defined with a suffix:
7. Multi (or Compound) Entries

\multiglossaryentry[firstsuffix=botulism]{cbot}{clostridium,botulinum}
\multiglossaryentry[firstsuffix=gas gangrene]{cperf}{clostridium,perfringens}
\multiglossaryentry[firstsuffix=tetanus]{ctet}{clostridium,tetani}

The rest of the document is as in §7.1.1.

Example 126: Multi-entries: hierarchical with first-use suffix

First use: Clostridium botulinum (botulism), C. perfringens (gas gangrene), C. tetani (tetanus).
Next use: C. botulinum, C. perfringens, C. tetani.

Glossary

C

Clostridium

botulinum 1

perfringens 1
tetani 1

7.1.3. Example: Category Suffix

This is an alternative to the previous example. Instead of storing the extra information in the \texttt{firstsuffix} key, the information is stored in the \texttt{user1} key of the last element (the species). A category suffix is used to look up the field and append it.

\newglossaryentry{botulinum}{name={botulinum},category={species},
user1={botulism},
description={},parent={clostridium}}
\newglossaryentry{perfringens}{name={perfringens},category={species},
user1={gas gangrene},
description={},parent={clostridium}}
\newglossaryentry{tetani}{name={tetani},category={species},
user1={tetanus},
description={},parent={clostridium}}
\mglssdefcategorysuffix{bacteria}{%
7. Multi (or Compound) Entries

\multiglossaryentry[category=bacteria]{cbot}{clostridium,botulinum}
\multiglossaryentry[category=bacteria]{cperf}{clostridium,perfringens}
\multiglossaryentry[category=bacteria]{ctet}{clostridium,tetani}

The result is the same as the previous example.

Example 127: Multi-entries: hierarchical with category suffix

First use: Clostridium botulinum (botulism), C. perfringens (gas gangrene), C. tetani (tetanus).
Next use: C. botulinum, C. perfringens, C. tetani.

Glossary

C

Clostridium

botulinum 1

perfringens 1

tetani 1

7.1.4. Example: Separators

The first example (§7.1.1) can be modified so that the species are also abbreviations. In this case, the separators are modified to suppress the space (\relax) if both the genus and species are abbreviated, or to use a non-breaking space (~) between the genus short form (shown on subsequent use) and the species long form (shown on first use). If the genus is showing the long form (first use) then a normal space is used.

Note that the separator attributes apply to the category of the element before the separator (not to the multi-entry category).
7. Multi (or Compound) Entries

\setabbreviationstyle[species]{long-only-short-only-desc}

\newabbreviation[category={species},
    description={},parent={clostridium}] {botulinum}{bot.} {botulinum}
\newabbreviation[category={species},
    description={},parent={clostridium}] {perfringens}{per.} {perfringens}
\newabbreviation[category={species},
    description={},parent={clostridium}] {tetani}{tet.} {tetani}

This will cause a double dot at the end of the second sentence, which can be suppressed using
the discardperiod and retainfirstuseperiod attributes.

\glssetcategoriesattributes{species}{discardperiod,retainfirstuse-period}{true}

This works because the final element’s post-link hook is transferred to the multi-entry post-
link hook, which can detect the sentence terminating period. If the post-link hook settings
are changed, for example, to postlinks=all, mpostlink=false then the feature won’t
work as the final element’s post-link hook can’t detect the period (because \gls is embedded
too deeply inside the internal workings of \mgls).

Example 128: Multi-entries: separators

First use: *Clostridium botulinum, C. perfringens, C. tetani.*
Next use: *C.bot., C.per., C.tet.*

Glossary

C

Clostridium 1

botulinum 1 1

perfringens 1

tetani 1

7.1.5. Example: Skipping Elements (Fragment Element)

This example is an alternative way of dealing with nested links (see §5.4).
\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage[stylemods,style=long]{glossaries-extra}
\makeglossaries
\setabbreviationstyle{long-short-sc}
\newabbreviation{ssi}{ssi}{server-side includes}
\newabbreviation{html}{html}{hypertext markup language}
\setabbreviationstyle{long-only-short-sc-only}
\newabbreviation[category={combinedabbrv},
\description={\glsxtrshort{ssi} enabled \glsxtrshort{html}}]{shtml-frag}{shtml}{enabled}
\glssetcategoryattribute{combinedabbrv}{multioptions}
\{usedskipothers=,firstsuffix=\glsxtrshort{\mglslastmainlabel}\}
\multiglossaryentry
\{category=combinedabbrv\}
\{shtml\}[shtml-frag]{ssi,shtml-frag,html}
\begin{document}
\begin{itemize}
\item Individual elements first use: \texttt{\gls{ssi}} and \texttt{\glshtml}.
\item Individual elements next use: \texttt{\gls{ssi}} and \texttt{\gls{html}}.
\item Multi-entry first use: \texttt{\mglsshtml}.
\item Multi-entry next use: \texttt{\mglsshtml}.
\item Resetting all\texttt{\glsresetall}\texttt{\glsreset{shtml}}:
\item Multi-entry first use: \texttt{\mglsshtml}.
\item Multi-entry next use: \texttt{\mglsshtml}.
\item Individual elements: \texttt{\gls{ssi}} and \texttt{\gls{html}}.
\end{itemize}
\end{document}
This uses the `multioptions` attribute to skip “other” elements on subsequent use. The problematic abbreviation (SHTML) is defined as a fragment that simply expands to “enabled” on first use. Note that the description has to be supplied for the glossary. The resulting document is shown below.

**Example 129: Multi-entries: skipping elements**

<table>
<thead>
<tr>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual elements first use: server-side includes (SSI) and hypertext markup language (HTML).</td>
<td>server-side includes (SSI) and hypertext markup language (HTML).</td>
</tr>
<tr>
<td>Individual elements next use: SSI and HTML.</td>
<td>SSI and HTML.</td>
</tr>
<tr>
<td>Multi-entry first use: SSI enabled HTML (SHTML).</td>
<td>SSI enabled HTML (SHTML).</td>
</tr>
<tr>
<td>Multi-entry next use: SHTML.</td>
<td>SHTML.</td>
</tr>
<tr>
<td>Resetting all:</td>
<td></td>
</tr>
<tr>
<td>Multi-entry first use: server-side includes (SSI) enabled hypertext markup language (HTML) (SHTML).</td>
<td>server-side includes (SSI) enabled hypertext markup language (HTML) (SHTML).</td>
</tr>
<tr>
<td>Multi-entry next use: SHTML.</td>
<td>SHTML.</td>
</tr>
<tr>
<td>Individual elements: SSI and HTML.</td>
<td>SSI and HTML.</td>
</tr>
</tbody>
</table>

**Glossary**

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML</td>
<td>hypertext markup language 1</td>
</tr>
<tr>
<td>SHTML</td>
<td>SSI enabled HTML 1</td>
</tr>
<tr>
<td>SSI</td>
<td>server-side includes 1</td>
</tr>
</tbody>
</table>

The key difference here from the example using \glsps at the end of §5.4 is that the individual elements hyperlink to their respective entries in the glossary on first use of \mglss. The problem is that with the colorlinks package option, it’s not obvious where the hyperlinks start and end. The suffix (SHTML) will hyperlink to the “shtml” entry in the glossary, so the “enabled” hyperlink is redundant. The simplest fix for this is to add `hyper=notmainfirst` to the option list, which will prevent “enabled” from being a hyperlink.

Another problem occurs where \mglss is used before the individual elements are used, which leads to their full expansion with a confusing amount of parentheses. A simple solution is to use the option `mglsopts=unsetothers`, which will unset the other (not-main) elements first. This can be localised with `presetlocal` but \mglss will then unset the first use flag globally, which means that the other elements won’t show the full form when they are first used on their on after \mglss. This can be switched to a local unset with `others=local`.

The complete set of options are now:
7. Multi (or Compound) Entries

Example 130: Multi-entries: skipping elements (unsettng others)

Individual elements first use: server-side includes (SSI) and hypertext markup language (HTML).

Individual elements next use: SSI and HTML.

Multi-entry first use: firstsuffix, SSI enabled HTML.

Multi-entry next use: firstsuffix, SSI SHTML HTML.

Resetting all:

Multi-entry first use: firstsuffix, SSI enabled HTML.

Multi-entry next use: firstsuffix, SSI SHTML HTML.

Individual elements: server-side includes (SSI) and hypertext markup language (HTML).

Glossary

<table>
<thead>
<tr>
<th>HTML</th>
<th>hypertext markup language 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHTML</td>
<td>SSI enabled HTML 1</td>
</tr>
<tr>
<td>SSI</td>
<td>server-side includes 1</td>
</tr>
</tbody>
</table>

This method still has two main drawbacks: the description must be added manually and the long form can’t be accessed with \glsxtrlong. The next example provides an alternative approach.

7.1.6. Example: Skipping Elements (Prefix and Post-Link Hooks)

This is a modified version of the previous example. In this case, the main element isn’t a fragment and also happens to have the same label as the multi-entry set. (\mgs{shtml} references the multi-entry label and \gls{shtml} references the individual entry.)

In this case, the nested parts are marked up with custom commands:
7. Multi (or Compound) Entries

\newrobustcmd{\combinedpre}[1]{\glsxps{#1}}
\newrobustcmd{\combinedpost}[1]{\glsxps{#1}}
\newabbreviation{shtml}{shtml}

{{}\combinedpre{ssi} enabled \combinedpost{html}}

This means that it’s no longer necessary to manually insert the description and the long form can be accessed as usual with \glsxtrshort{shtml}. Note that it is necessary to define the custom commands robustly otherwise they will need to be protected against premature expansion:

\newcommand{\combinedpre}[1]{\glsxps{#1}}
\newcommand{\combinedpost}[1]{\glsxps{#1}}
\newabbreviation{shtml}{shtml}

{{}\protect\combinedpre{ssi} enabled \protect\combinedpost{html}}

In both cases, an initial empty group is added to guard against any sentence case commands, such as \Glsxtrlong.

The abbreviations all use the long-postshort-sc-user style, which places the short form in the post-link hook on first use. The \gls post-link hook for the main element can be transferred to the \mgls post-link using:

\texttt{mpostlinkelement=main}

All elements have their individual post-link hooks suppressed by default. As in the previous example, the other elements can be skipped on subsequent use:

\texttt{usedskipothers}

Within \mgls, the nested content needs to be suppressed, which can be done by redefining the custom commands. This can be done in the multi-entry prefix. Since the entire content of \mgls (except for the final multi-entry post-link hook) occurs inside a group, this redefinition will be localised.

The complete document is as follows:
7. Multi (or Compound) Entries

\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage[stylemods,style=long]{glossaries-extra}
\makeglossaries
\setabbreviationstyle{long-postshort-sc-user}
newabbreviation{ssi}{ssi}{server-side includes}
newabbreviation{html}{html}{hypertext markup language}
newrobustcmd{\combinedpre}[1]{\glsps{#1}}
newrobustcmd{\combinedpost}[1]{\glsps{#1}}
newabbreviation{shtml}{shtml}
{\combinedpre{shtml} enabled \combinedpost{html}}
glssetcategoryattribute{combinedabbrv}{multioptions}
{\%
\renewcommand{\combinedpre}[1]{\ignorespaces}
\renewcommand{\combinedpost}[1]{\unskip}
\}
multiglossaryentry
\{category=combinedabbrv\}
{shtml}{shtml,ssi,shtml,html}
mglssdefcategoryprefix{combinedabbrv}{\%
\renewcommand{\combinedpre}[1]{\ignorespaces%%
\renewcommand{\combinedpost}[1]{\unskip%%}
\}
\begin{document}
Individual elements first use: \gls{ssi} and \gls{html}.
Individual elements next use: \gls{ssi} and \gls{html}.
Multi-entry first use: \mgls{shtml}.
Multi-entry next use: \mgls{shtml}.
\end{document}
7. Multi (or Compound) Entries

Individual entry first use: `\gls{shtml}`.

Resetting all `\glsresetall\mglssresetall`:

Multi-entry first use: `\mglss{shtml}`.

Multi-entry next use: `\mglss{shtml}`.

Individual elements: `\gls{ssi} and \gls{html}`.

Resetting all `\glsresetall\mglssresetall`:

Individual entry first use: `\gls{shtml}`.

Multi-entry first use: `\mglss{shtml}`. (Wrong!)

\printglossaries
\end{document}

This now produces:
Example 131: Multi-entries: skipping elements (prefix and post-link hooks)

   Individual elements first use: server-side includes (SSI) and hypertext markup language (HTML).
   Individual elements next use: SSI and HTML.
   Multi-entry first use: SSI enabled HTML (SHTML).
   Multi-entry next use: SHTML.
   Individual entry first use: SHTML.
   Resetting all:
   Multi-entry first use: server-side includes enabled hypertext markup language (SHTML).
   Multi-entry next use: SHTML.
   Individual elements: SSI and HTML.
   Resetting all:
   Individual entry first use: SSI enabled HTML (SHTML).
   Multi-entry first use: server-side includes SHTML hypertext markup language. (Wrong!)

**Glossary**

<table>
<thead>
<tr>
<th>HTML</th>
<th>hypertext markup language 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHTML</td>
<td>SSI enabled HTML 1</td>
</tr>
<tr>
<td>SSI</td>
<td>server-side includes 1</td>
</tr>
</tbody>
</table>

Note the last two paragraphs, which highlights what happens if \texttt{\textbackslash gls\{shtml\}} is used before \texttt{\textbackslash mgl\{shtml\}} when neither of the other elements (ssi and html) have been used. The final instance of \texttt{\textbackslash mgl\{} has produced the wrong result. This is because it’s the first use of the multi-entry shtml but not the first use of the individual entry shtml.

One way around this is to modify the prefix to ensure that the main element’s first use flag matches the multi-entry’s first use flag:

```latex
\mgl\{categoryprefix\{combinedabbrv\}\{\%
\renewcommand\{\combinedpre\}[1]\{\ignorespaces\%
\renewcommand\{\combinedpost\}[1]\{\unskip\%
\mgl\{isfirstuse\}
\{\mgllocalreset\{\mglcurrentmainlabel\}\}\%
\{\mgllocalunset\{\mglcurrentmainlabel\}\}%
\}
```
7. Multi (or Compound) Entries

The `showtargets=annoteleft` option can be used to mark up the links with the targets. For example, the first instance of `\mgl{shtml}` will show as:

```
```

Each entry has an individual hyperlink to its own glossary item, which may be confusing. This can be made clearer by suppressing the main element link on first use with:

```
hyper=notmainfirst
```

(as in the previous example), and adjusting the abbreviation style so that the parenthetical content in the post-link hook has a hyperlink:

```
\renewcommand*{\glsxtruserparen}[2]{% 
  \glsxtrfullsep{#2}% 
  \glsxtrparen 
  {\glshyperlink[#1]{#2}% 
   \ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{}} 
  % 
  }% 
}
```

The remaining problem is how to deal with the possibility that `\mgl{shtml}` may come before the first use of the other elements. For example:

```
Multi-entry first use: \mgl{shtml}.
Individual elements: \gls{ssi} and \gls{html}.
```

This leads to:

```
Multi-entry first use: server-side includes enabled hypertext markup language (SHTML).
Individual elements: SSI and HTML.
```

This means that the abbreviations SSI and HTML aren’t explained in the document text. One way around this is to only locally unset the other element first use flags:
7. Multi (or Compound) Entries

With the above setting, the following:

- Multi-entry first use: `\gls{shtml}`.
- Multi-entry next use: `\gls{shtml}`.
- Individual elements: `\gls{ssi}` and `\gls{html}`.

will now produce:

- Multi-entry first use: server-side includes enabled hypertext markup language (shtml).
- Multi-entry next use: shtml.
- Individual elements: server-side includes (ssi) and hypertext markup language (html).

### 7.2. Main and Other Elements

The list of labels provided in the final argument of `\multiglossaryentry` consists of a main element and all the other elements. If the main element isn’t identified in the optional argument, it’s assumed to be the final element in the list.

The main element allows you to determine which target to use if you want the entire content of `\gls` to be a single hyperlink. You can also use the settings described in §7.9 to only index the main element.

You can change the main element using: `\glsSetMain`. The new main label provided in the second argument must be in the list corresponding to `<multi-label>`. This change is locally applied to the current scope. Note that if you are using `bib2gls`, this change in the document can’t be detected.

The main element can also be used to identify which element should be displayed in the plural with `\gls{mainpl}`. For example:
In the above, two multi-entries are defined: greatgrebe and littlegrebe. In both cases the main element is grebe (the last element). Using \mglsp will show the plural for all elements, but using \mglsmainpl will only use the plural for the main element (grebe). For example:

Plural all: \mglsp{greatgrebe}, \mglsp{littlegrebe}.
Plural main: \mglsmainpl{greatgrebe}, \mglsmainpl{littlegrebe}.

produces:


7.3. Prefixes and Suffixes

A multi-entry may have associated prefixes and suffixes. These are scoped and are placed outside of the hyperlinks and encapsulating commands. They are not affected by case-changing commands, such as \Mgls. If you want a prefix to obey case-changing, use the \mpgls-like commands instead (§7.11.4).

The prefix is inserted with:

\mglsprefix

The default definition is:
This will insert the current prefix unless there is prefix command associated with the current category.

The suffix is inserted with:

\mglssuffix

This command is defined as follows:

\newcommand*{\mglssuffix}{%
  \ifdefempty{\mglscurrentcategory}{%}
    {%}
      \mglshascategorysuffix{\mglscurrentcategory}%
        \mglsusecategorysuffix{\mglscurrentcategory}%
      \mglscurrentprefix%
    %}
%}

If there is a suffix associated with the current category, that will be used, otherwise if the current suffix isn’t empty this inserts a space followed by the current suffix in parentheses.

You can access the label of the last (non-skipped) element with \mglslastelementlabel.

Note that in both cases the category corresponds to the multi-entry category (see §7.8).

To define a category-dependent prefix, use:

\mglsqdefcategoryprefix{\langle category-label \rangle}{\langle definition \rangle}

You can reference the current prefix with \mglscurrentprefix within \langle definition \rangle.

To define a category-dependent suffix, use:

\mglsqdefcategoriesuffix{\langle category-label \rangle}{\langle definition \rangle}
7. Multi (or Compound) Entries

You can reference the current suffix with \mglscurrentsuffix within \emph{definition}.

The default definition of \mglsprefix tests if there is a category prefix using:

\mglshascategoryprefix{\langle category-label \rangle}{\langle true \rangle}{\langle false \rangle}

This does \emph{true} if a prefix has been assigned to the given category, otherwise it does \emph{false}.

If you need to obtain the prefix for a particular category, you can use:

\mglssusecategoryprefix{\langle category-label \rangle}

This expands to the prefix, if set, for the given category or to nothing otherwise.

The default definition of \mglssuffix tests if there is a category suffix using:

\mglshascategorysuffix{\langle category-label \rangle}{\langle true \rangle}{\langle false \rangle}

This does \emph{true} if a suffix has been assigned to the given category, otherwise it does \emph{false}.

If you need to obtain the suffix for a particular category, you can use:

\mglssusecategorysuffix{\langle category-label \rangle}

This expands to the suffix, if set, for the given category or to nothing otherwise.

The current prefix \mglscurrentprefix and \mglscurrentsuffix are obtained as follows:

- if this is the first use of the multi-entry (§7.7) then the \emph{prefix} is set to the value of the firstprefix option and the \emph{suffix} is set to the value of the firstsuffix option;
- otherwise the \emph{prefix} is set to the value of the usedprefix option and the \emph{suffix} is set to the value of the usedsuffix option.

The prefix and suffix (if set) are placed outside of the hyperlink and text formatting encapsulator. They are not affected by case-changing commands such as \texttt{Mgls} or \texttt{MGLS}.

For example:

\setabbreviationstyle{long-only-short-only}
\newabbreviation{clostridium}{C.}\{Clostridium\}
\newglossaryentry{botulinum}{name={botulinum},description={}}
7. Multi (or Compound) Entries

\multiglossaryentry[firstsuffix=botulism]
\{cbot\}\{clostridium, botulinum\}

On first use, this produces (assuming the “clostridum” element hasn’t been used previously):

Clostridium botulinum (botulism).

On subsequent use, this produces:

C. botulinum.

7.4. Separators

The separators between each instance of \gls are given by the following commands, which all take two arguments. The first argument is the label of the previous element. The second argument is the label of the following element.

\glscombinedsep{⟨prev label⟩}{⟨next label⟩}

This is inserted between two entries that have both been marked as used. The default definition is:

\newcommand*{\glscombinedsep}[2]{% 
  \glshasattribute{#1}{combinedsep}% 
  \glsgetattribute{#1}{combinedsep}% 
  { }%}

This will use the combinedsep attribute for the ⟨prev label⟩’s category, if set. Otherwise this just does a space. Note that this ignores the second argument.

\glscombinedfirstsep{⟨prev label⟩}{⟨next label⟩}

This is inserted between two entries where only the next entry has been marked as used. The default definition is:
7. Multi (or Compound) Entries

\newcommand*{\glscombinedfirstsep}[2]{% 
  \glshasattribute{#1}{combinedfirstsep}\% 
  {\glsgetattribute{#1}{combinedfirstsep}}\% 
  {\glscombinedsep{#1}{#2}}\% 
}\n
This will use the combinedfirstsep attribute for ⟨prev label⟩’s category, if set. If that attribute isn’t set, \glscombinedsep is used.

\glscombinedsepfirst{⟨prev label⟩}{⟨next label⟩}

This is inserted between two entries where only the previous entry has been marked as used. The default definition is:

\newcommand*{\glscombinedsepfirst}[2]{% 
  \glshasattribute{#1}{combinedsepfirst}\% 
  {\glsgetattribute{#1}{combinedsepfirst}}\% 
  {\glscombinedsep{#1}{#2}}\% 
}\n
This will use the combinedsepfirst attribute for ⟨prev label⟩’s category, if set. If that attribute isn’t set, \glscombinedsep is used.

\glscombinedfirstsepfirst{⟨prev label⟩}{⟨next label⟩}

This is inserted between two entries where both have been marked as used. The default definition is:

\newcommand*{\glscombinedfirstsepfirst}[2]{% 
  \glshasattribute{#1}{combinedfirstsepfirst}\% 
  {\glsgetattribute{#1}{combinedfirstsepfirst}}\% 
  {\glscombinedsep{#1}{#2}}\% 
}\n
This will use the combinedfirstsepfirst attribute for ⟨prev label⟩’s category, if set. If that attribute isn’t set, \glscombinedsep is used.

These commands may be redefined as required. For example, to have no space between two elements that have both been marked as used and are both abbreviations (disregarding category attributes):

344
7. Multi (or Compound) Entries

There are some commands for redefining the above separators to common combinations.

\glssetcombinedsepabbrvnbsp

This does the following:

This uses a non-breaking space (\textendash) following an abbreviation (that has already been marked as used). Note that if the associated attributes are set the commands will behave according to the attribute.

\glssetcombinedsepabbrvnone
7. Multi (or Compound) Entries

\begin{verbatim}
\renewcommand*{\glscombinedsep}[2]{% 
grishasattribute{#1}{combinedsep}\% 
{\glsgetattribute{#1}{combinedsep}}\% 
{\ifglshasshort{#1}{\{}{\ifglshasshort{#2}{\{}{ }}\}}}\% 
\renewcommand*{\glscombinedsepfirst}[2]{% 
grishasattribute{#1}{combinedsepfirst}\% 
{\glsgetattribute{#1}{combinedsepfirst}}\% 
{\ifglshasshort{#1}{\{}{ }}\% 
\renewcommand*{\glscombinedfirstsep}[2]{% 
grishasattribute{#1}{combinedfirstsep}\% 
{\glsgetattribute{#1}{combinedfirstsep}}\% 
{\ifglshasshort{#2}{\{}{ }}\% 
\renewcommand*{\glscombinedfirstsepfirst}[2]{% 
grishasattribute{#1}{combinedfirstsepfirst}\% 
{\glsgetattribute{#1}{combinedfirstsepfirst}}\% 
{ }\% 
\end{verbatim}

This does nothing if either element are abbreviations that have already been used. Note that if the associated attributes are set the commands will behave according to the attribute.

\begin{verbatim}
\glssetcombinedsepnarrow{⟨width⟩}{⟨narrow-sep⟩}
\end{verbatim}

This is rather more complicated as it measures a field value and uses ⟨narrow-sep⟩ if the width is less than ⟨width⟩. The field value is determined as follows:

- on first use the long field is used if it is set otherwise the first field is used;
- otherwise the short field is used if it is set otherwise the text field is used;

Note that this doesn’t take into account fonts, hooks, abbreviation styles or plural forms (e.g. \mglspl) or other field references (e.g. \glsname). If the associated attributes are set the commands will behave according to the attribute.
7. Multi (or Compound) Entries

7.5. \mgl s Element Hooks

The \mgl s-like commands use the following hooks:

\mglselementprehook

This is done before each (non-skipped) element. (Default does nothing.)

\mglselementposthook

This is done after each (non-skipped) element. (Default does nothing.) Note that this is different from the normal entry post-link hook \glspostlinkhook. If the individual entry post-link hook is enabled (see the postlinks key in §7.9), this will go before \mglselement- posthook.

The definitions of the following commands are scoped within the \mgl s-like commands so they can’t be accessed elsewhere (including in the post-link hook, see §7.6). They may be used in the above hooks or in the separator commands (described in §7.4) or in the command used to encapsulate the entire content. They can also be used in the post-link hook (see §5.5) to determine if an entry is being used within a \mgl s-like command.

\mglscurrentmultilabel

Expands to the multi-entry label.

\mglscurrentmainlabel

Expands to the label of the main element.

\mglscurrentlist

Expands to the complete comma-separated list of elements.

\mglscurrentoptions

Expands to the options used when the multi-entry was defined. This doesn’t include options set with \multiglossaryentrysetup or those passed to \mgl s (or whichever variant is
being used).

\texttt{\textbackslash mglscurrentcategory}

Expands to the multi-entry category current in effect.

\texttt{\textbackslash glsxtrcurrentmglscsname}

Expands to the control sequence name of the calling command (for example, mgl\textit{s} or mgl\textit{spl}). To test if the current multi-entry is the first use:

\texttt{\textbackslash mglsisfirstuse\{true\}\{false\}}

This does \textit{(true)} if this is the first use otherwise it does \textit{(false)}. Note that this applies to the multi-entry first use flag not the first use flags of the individual elements.

At each iteration of the loop over the element list, the following commands are set, which can be accessed in hooks such as \texttt{\textbackslash mglselementprehook} or in hooks used by the underlying \texttt{\textbackslash gls} etc commands. For example, if \texttt{\textbackslash mglscurrentlabel} is defined then \texttt{\textbackslash gls} is being used inside \texttt{\textbackslash mgl\textit{s}}.

\texttt{\textbackslash mglscurrentlabel}

Expands to the current element label.

\texttt{\textbackslash mglselementindex}

This is a count register that is set to the element index.

\texttt{\textbackslash mglscurrentprefix}

Expands to the current multi-entry prefix.

\texttt{\textbackslash mglscurrentsuffix}
7. Multi (or Compound) Entries

Expands to the current multi-entry suffix.

\mglslast{\langle true \rangle}{\langle false \rangle}

If this is the last iteration, does ⟨true⟩ otherwise does ⟨false⟩. This takes the skip options into account, so the last iteration may not necessarily be when the element index is equal to the total number of elements.

7.6. Post-Link Hook

There is a hook that occurs at the end the \mglslike commands according to the mpostlink setting (see §7.9). The hook used depends on the mpostlinkelement option. These hooks can’t access the commands described in §7.5 as the hook occurs outside of the scope in which they are defined.

The mpostlinkelement=custom option uses:

\mglscustompostlinkhook

This does nothing by default.

The mpostlinkelement=last option uses:

\mglslastelementpostlinkhook

which emulates the post-link hook of the last element.

The mpostlinkelement=main option uses:

\mglslastmainpostlinkhook

which emulates the post-link hook of the main element.

The default settings postlinks=none, mpostlink=true, mpostlinkelement=last will suppress the individual element post-link hooks (\glspostlinkhook) and do the multi-entry post-link hook for the last element (\mglslastelementpostlinkhook).

If you have the final element’s post-link hook enabled and the multi-entry post-link hook enabled (for example, postlinks=all, mpostlink=true, mpostlinkelement=last), the final element’s post-link hook will be done twice. Similarly for the main element with postlinks=all, mpostlink=true, mpostlinkelement=main.

The following commands are available for use in these hooks and may also be used in the
7. Multi (or Compound) Entries

definition of \mglssuffix.

\mglslastmultilabel
Expands to the multi-entry label.

\mglslastcategory
Expands to the multi-entry category (see §7.8). This will be empty if no category was assigned.

\mglswasfirstuse{⟨true⟩}{⟨false⟩}
If that was the first use of the multi-entry (see §7.7) this does ⟨true⟩ otherwise it does ⟨false⟩.

7.6.1. Last Element

The following commands relate to the last element.

\mglslastelementlabel
Expands to the label of the last non-skipped element. If all elements were skipped or if the multi-entry wasn’t defined, this will be empty.

Test if the last element was skipped:

\mglsisiflastelements skipped{⟨true⟩}{⟨false⟩}
If the last element was skipped this does ⟨true⟩ otherwise it does ⟨false⟩. If all elements were skipped or if the multi-entry wasn’t defined, this will do ⟨true⟩.

Test if the last element was its first use:

\mglsisiflastelementwasfirstuse{⟨true⟩}{⟨false⟩}
If the last non-skipped element was used for the first time this does ⟨true⟩ otherwise it does ⟨false⟩. (Corresponds to \glsxtrifwasfirstuse.) If all elements were skipped or if the multi-entry wasn’t defined, this will do ⟨true⟩.
7. Multi (or Compound) Entries

Test if the last element was plural:

\mglslastelementwasplural\{true\}\{false\}

If the last non-skipped element had the plural form displayed, this does \textit{true} otherwise it does \textit{false}. (Corresponds to \texttt{\glsifplural}.) If all elements were skipped or if the multi-entry wasn’t defined, this will do \textit{false}.

Test if the last element was had any case-changing applied:

\mglslastelementcapscase\{no-change\}\{firstuc\}\{all caps\}

Corresponds to \texttt{\glscapscase} of the last non-skipped element. If all elements were skipped or if the multi-entry wasn’t defined, this will do \textit{no-change}.

7.6.2. Main Element

The following commands relate to the main element.

\mglslastmainlabel

Expands to the label of the main element from the multi-entry that was just referenced. If the main element was skipped or if the multi-entry wasn’t defined, this will be empty. If this is the same as \texttt{\mglslastelementlabel} then the main element was the last element.

Test if the main element was skipped:

\mglslastmainskipped\{true\}\{false\}

If the main element from the multi-entry that was just referenced was skipped this does \textit{true} otherwise it does \textit{false}. If the multi-entry wasn’t defined, this will do \textit{true}.

Test if the main element was its first use:

\mglslastmainwasfirstuse\{true\}\{false\}

If the main element was used for the first time this does \textit{true} otherwise it does \textit{false}. (Corresponds to \texttt{\glsxtrifwasfirstuse}.) If the main element was skipped or if the multi-entry wasn’t defined, this will do \textit{true}.
7. Multi (or Compound) Entries

Test if the main element was plural:

\mgsiflastmainwasplural{⟨true⟩}{⟨false⟩}

If the main element from the multi-entry that was just referenced had its plural form displayed this does {⟨true⟩} otherwise it does {⟨false⟩}. (Corresponds to \glsifplural.) If the main element was skipped or if the multi-entry wasn’t defined, this will do {⟨false⟩}.

Test if the main element was had any case-changing applied:

\mgsiflastmaincapscase{⟨no-change⟩}{⟨firstuc⟩}{⟨all caps⟩}

Corresponds to \glsccapscase of the main element from the multi-entry that was just referenced. If the main element was skipped or if the multi-entry wasn’t defined, this will do {⟨no-change⟩}.

7.7. Multi-Entry First Use

Each multi-entry set has an associated first use flag. This is independent of the first use flag associated with the individual entries that make up the set. As with the \gls-like commands, \mgs unsets this flag. Unlike the \glistext-like commands, all the commands described in §7.11 (including commands like \mgsname) unset this flag, even if the elements use commands like \glsname that don’t unset the entry’s first use flag.

You can determine whether or not a multi-entry set has been marked as used with:

\ifmgsused{⟨multi-label⟩}{⟨true⟩}{⟨false⟩}

This does {⟨true⟩} if the given multi-entry has been marked as used, otherwise it does {⟨false⟩}.

You can (globally) unset the flag (mark the set as having been referenced) with:

\mgsunset{⟨multi-label⟩}

or reset it with:

\mgsreset{⟨multi-label⟩}
There are also local versions of these commands:

\mgl泷localunset{⟨multi-label⟩}

which locally unsets the flag and

\mgl泷localreset{⟨multi-label⟩}

which locally resets the flag.

It’s also possible to unset or reset all multi-entries.

\mgl泷unsetall

Unsets all multi-entries.

\mgl泷resetall

Resets all multi-entries.

Note that unsetting or resetting any of the individual element first use flags doesn’t affect the multi-entry flag. Similarly, unsetting or resetting the multi-entry flag doesn’t affect the first use flags of the individual elements.

### 7.8. Multi-Entry Category

A multi-entry set may have an associated category set using the category key described in §7.9. This isn’t set by default, but if it is set the category may have attributes set in the usual way. The multi-entry category is independent of the individual entry categories. The following attribute is recognised by commands like \mgls:

\texttt{\textbackslash multioptions} = ⟨options⟩

The value are the default options to apply to any multi-entry set with the given category. These can be overridden by the first optional argument of \multiglossaryentry or by the
setup key in the first optional argument of commands like `\mgls`.

\begin{itemize}
  \item \texttt{\textbackslash combinedfirstsep=⟨separator⟩}
  \textit{The separator to use for `\glscombinedfirstsep`.}
  \item \texttt{\textbackslash combinedfirstsepfirst=⟨separator⟩}
  \textit{The separator to use for `\glscombinedfirstsepfirst`.}
  \item \texttt{\textbackslash combinedsepfirst=⟨separator⟩}
  \textit{The separator to use for `\glscombinedsepfirst`.}
  \item \texttt{\textbackslash combinedsep=⟨separator⟩}
  \textit{The separator to use for `\glscombinedsep`.}
\end{itemize}

Note that you can’t access the category or its attributes via the multi-entry label (for example, with `\glshasattribute`). If you need to access the current multi-entry’s category within any of the `\mgls`-like hooks (§7.5), you can obtain the category with `\mglscurrentcategory` and use commands like `\glshascategoryattribute`.

### 7.9. Multi-Entry Settings

The settings that govern all multi-entries can be set using:

\begin{itemize}
  \item `\multiglossaryentrysetup{⟨options⟩}`
\end{itemize}

The `⟨options⟩` are the same as for `\multiglossaryentry`.

Whenever the `\mgls`-like commands are used, options are applied in the following order:

1. general options identified by `\multiglossaryentrysetup`;
2. the category key is assigned if it’s in the general options, `\multiglossaryentry` or `setup` key;
3. `multioptions` category attribute (if set);
4. any options provided in the first optional argument of `\multiglossaryentry`;

354
7. Multi (or Compound) Entries

5. any options provided in the setup key in the first optional argument of the \mgls-like command.

These options are described below.

### 7.9.1. Indexing

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>indexmain=⟨value⟩</td>
<td>default: true; initial: true</td>
</tr>
<tr>
<td>indexothers=⟨value⟩</td>
<td>default: true; initial: true</td>
</tr>
</tbody>
</table>

This setting may take one of the following values:
- **false**: don’t index the main entry;
- **true**: index the main entry;
- **first**: only index the main entry if it’s the first use (of the main entry).

### 7.9.2. Location Formats (Encaps)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encapmain=⟨value⟩</td>
<td>initial: glsnumberformat</td>
</tr>
<tr>
<td>encapothers=⟨value⟩</td>
<td>initial: glsnumberformat</td>
</tr>
</tbody>
</table>

This setting value should be the value to pass to format option (location encap) for the main entry.

This setting value should be the value to pass to format option (location encap) for the other (not main) entries.
7.9.3. Post-Link Hooks

\[
\text{postlinks} = \langle \text{value} \rangle \quad \text{initial: none}
\]

This setting determines whether or not to enable the individual element’s post-link hook. The value may be one of:

- **none** suppress the post-link hook for all elements;
- **all** don’t suppress the post-link hook for all elements;
- **notlast** only suppress the post-link hook for the last element;
- **mainnotlast** suppress the post-link hook for all “other” (not main) elements and for the last element (so only the main element will have its post-link hook as long as it’s not the last element);
- **mainonly** suppress the post-link hook for all “other” (not main) elements;
- **othernotlast** suppress the post-link hook for the main element and for the last element (so only the other elements will have their post-link hook as long as the element isn’t the last one);
- **otheronly** suppress the post-link hook for the main element.

\[
\text{mpostlink} = \langle \text{value} \rangle \quad \text{default: true; initial: true}
\]

This setting determines whether or not to enable the multi-entry post-link hook (see §7.6). The value may be one of:

- **false** suppress the multi-entry post-link hook;
- **true** enable the multi-entry post-link hook;
- **firstonly** enable the multi-entry post-link hook only for the first use of the multi-entry;
- **usedonly** enable the multi-entry post-link hook only for the subsequent use of the multi-entry.

\[
\text{mpostlinkelement} = \langle \text{value} \rangle \quad \text{initial: last}
\]

This setting indicates which post-link hook should be used if the multi-entry post-link hook has been enabled. Allowed values:
7. Multi (or Compound) Entries

last use \mglslastelementpostlinkhook (that is, use the post-link hook for the last element);

main use \mglslastmainpostlinkhook (that is, use the post-link hook for the main element);

custom use \mglscustompostlinkhook.

Some combinations may cause a repeated hook.

7.9.4. Prefixes and Suffixes

See §7.3 for more information on prefixes and suffixes. Note that the prefixes and suffixes are not affected by case-changing commands such as \Mgls or \MGLS. If you want a prefix to obey case-changing, use the \mpgls-like commands instead (see §7.11.4).

\begin{itemize}
\item \texttt{firstprefix=⟨value⟩}
\end{itemize}

The prefix to use on first use of the multi-entry.

\begin{itemize}
\item \texttt{usedprefix=⟨value⟩}
\end{itemize}

The prefix to use on subsequent use of the multi-entry.

\begin{itemize}
\item \texttt{firstsuffix=⟨value⟩}
\end{itemize}

The suffix to use on first use of the multi-entry.

\begin{itemize}
\item \texttt{usedsuffix=⟨value⟩}
\end{itemize}

The suffix to use on subsequent use of the multi-entry.

7.9.5. Skipping Elements

The skip options apply to the multi-entry first use flag not the individual element first use. See §7.7.
7. Multi (or Compound) Entries

\begin{itemize}
\item \texttt{firstskipmain} = \langle boolean \rangle \quad \text{default: true; initial: false}
\begin{itemize}
\item If \texttt{true}, the main element will be omitted on (multi-entry) first use.
\end{itemize}

\item \texttt{firstskipothers} = \langle boolean \rangle \quad \text{default: true; initial: false}
\begin{itemize}
\item If \texttt{true}, the other (non-main) elements will be omitted on (multi-entry) first use.
\end{itemize}

\item \texttt{usedskipmain} = \langle boolean \rangle \quad \text{default: true; initial: false}
\begin{itemize}
\item If \texttt{true}, the main element will be omitted on (multi-entry) subsequent use.
\end{itemize}

\item \texttt{usedskipothers} = \langle boolean \rangle \quad \text{default: true; initial: false}
\begin{itemize}
\item If \texttt{true}, the other (non-main) elements will be omitted on (multi-entry) subsequent use.
\end{itemize}
\end{itemize}

Note that it is technically possible to set the skip options so that both the main and the other elements are skipped. However, by default, this will generate a warning and only the final optional argument (the \langle \texttt{insert} \rangle) will be displayed. There won’t be a loop over all elements so the commands set at each iteration, such as \texttt{\mglscurrentlabel}, won’t be defined.

The warning and the insertion of the \langle \texttt{insert} \rangle is done by:

\begin{itemize}
\item \texttt{\glsxtrmglsWarnAllSkipped\{message\}\{insert\}\{fmt-cs\}}
\end{itemize}

where \langle message \rangle is the warning message and \langle cs \rangle is the control sequence that encapsulates the entire content (including hyperlink and the \texttt{textformat} setting, if enabled).

If, for some particular reason, you want this scenario, you can redefine this command to omit the warning.

7.9.6. General

\begin{itemize}
\item \texttt{hyper} = \langle value \rangle \quad \text{initial: individual}
\begin{itemize}
\item This setting may take one of the following values:
\item \texttt{none} no hyperlinks;
\item \texttt{allmain} encapsulate the entire content with a single hyperlink to the main entry’s target;
\item \texttt{mainonly} only hyperlink the main entry;
\end{itemize}
\end{itemize}
7. Multi (or Compound) Entries

individual  hyperlink each entry individually;
otheronly  only hyperlink the other entries;
notmainfirst  don’t hyperlink the main entry on multi-entry first use;
nototherfirst  don’t hyperlink the other entries on multi-entry first use;
notfirst  don’t hyperlink any entries on multi-entry first use.

textformat={value}  \( initial: \@\text{firstofone} \)

This setting value should be the control sequence name (without the leading backslash) of the command used to encapsulate the entire content.

category={category-label}

The category to apply to the multi-entry. This is independent of the categories of each of the elements. The default is no category. See §7.8.

mglsopts={option list}

Default options to pass to commands like \mgs\ls\s. Note that setup can’t be used within this value.

7.10. \mgs\ls\s Options

The \langle options\rangle for \mgs\ls\s (and similar commands) are listed below. Any additional options provided will be appended to the all value. For example:

\mgs\ls\s[\text{counter=chapter}]\{\text{cbot}\}

is equivalent to:

\mgs\ls\s[\text{all=counter=chapter}]\{\text{cbot}\}

Whereas:
7. Multi (or Compound) Entries

\texttt{\textbackslash mgl\textbackslash s[counter=chapter, all=counter=section]\{cbot\}}

is treated as:

\texttt{\textbackslash mgl\textbackslash s[all=counter=section, counter=chapter]\{cbot\}}

which has the same effect as:

\texttt{\textbackslash mgl\textbackslash s[all=counter=chapter]\{cbot\}}

The descriptions below reference \texttt{\textbackslash gls} which is used internally by \texttt{\textbackslash mgl\textbackslash s}. Replace this with \texttt{\textbackslash gls\texttt{pl}} etc as applicable for the variants, such as \texttt{\textbackslash mgl\textbackslash s\texttt{pl}}.

\texttt{\textbackslash setup\texttt{=}(option list)}

The value should be a list of any options that can be passed to \texttt{\textbackslash multiglossary\textbackslash entry\texttt{setup}}. These options will override any conflicting options that were supplied to \texttt{\textbackslash multiglossary\texttt{-entry}} or \texttt{\textbackslash multiglossary\textbackslash entry\texttt{setup}}. Note that \texttt{\textbackslash mgl\textbackslash s\texttt{opts}} can’t be used within this value.

\texttt{\textbackslash all\texttt{=}(option list)}

The value should be a list of any options that can be passed to \texttt{\textbackslash gls}. These options will be passed to each instance of \texttt{\textbackslash gls} and will override any conflicting setting in \texttt{\textbackslash setup}.

\texttt{\textbackslash main\texttt{=}(option list)}

The value should be a list of any options that can be passed to \texttt{\textbackslash gls}. These options will be passed to the instance of \texttt{\textbackslash gls} used for the main label and will override any conflicting setting in \texttt{\textbackslash all}.

\texttt{\textbackslash others\texttt{=}(option list)}

The value should be a list of any options that can be passed to \texttt{\textbackslash gls}. These options will be passed to each instance of \texttt{\textbackslash gls} used for the other (not main) labels and will override any
7. Multi (or Compound) Entries

conflicting setting in all.

\[\text{hyper}=\langle\text{boolean}\rangle\quad \text{default: true}\]

A boolean option to determine whether or not to use hyperlinks (if supported). This may cause a conflict with other options, but is essentially provided to allow the starred version of \text{\textbackslash mgl}s to switch off all hyperlinks.

\[\text{multiunset}=\langle\text{value}\rangle\quad \text{initial: global}\]

This only applies to the multi-entry first use flag, described in §7.7, not the first use flags of the elements. The value may be one of:

- \text{global} globally unset the flag;
- \text{local} locally unset the flag;
- \text{none} don’t unset the flag.

\[\text{presetlocal}=\langle\text{boolean}\rangle\quad \text{default: true; initial: false}\]

A boolean option that governs whether or not the following options should have a local or global effect.

\[\text{resetall}=\langle\text{boolean}\rangle\quad \text{default: true; initial: false}\]

A boolean option to determine whether or not to reset all elements \textit{before} using \text{\textbackslash gls}. This option refers to the individual entry’s first use flag not the multi-entry first use flag. (This is similar to passing \text{\textbackslash prereset} to each \text{\textbackslash gls} but it’s also applied to any skipped elements.)

\[\text{resetmain}=\langle\text{boolean}\rangle\quad \text{default: true; initial: false}\]

A boolean option to determine whether or not to reset the main entry’s first use flag \textit{before} using \text{\textbackslash gls}.

\[\text{resetothers}=\langle\text{boolean}\rangle\quad \text{default: true; initial: false}\]
A boolean option to determine whether or not to reset the first use flag of all the other (not main) elements *before* using `\gls`.

\begin{verbatim}
unsetall=(boolean) default: true; initial: false
\end{verbatim}

A boolean option to determine whether or not to unset all elements *before* using `\gls`. This option refers to the individual entry’s first use flag not the multi-entry first use flag. (This is similar to passing `preunset` to each `\gls` but it’s also applied to any skipped elements.)

\begin{verbatim}
unsetmain=(boolean) default: true; initial: false
\end{verbatim}

A boolean option to determine whether or not to unset the main entry’s first use flag *before* using `\gls`.

\begin{verbatim}
unsetothers=(boolean) default: true; initial: false
\end{verbatim}

A boolean option to determine whether or not to unset the first use flag of all the other (not main) elements *before* using `\gls`.

The `\glsreset{}` options all use:

\begin{verbatim}
\glsreset{⟨entry-label⟩}
\end{verbatim}

The unset... options all use:

\begin{verbatim}
\glsunset{⟨entry-label⟩}
\end{verbatim}

These take the `presetlocal` into account.

An alternative way of resetting the other elements is to use:

\begin{verbatim}
\glsunsetothers{⟨multi-label⟩}
\end{verbatim}

or for a local reset:

\begin{verbatim}
\glslocalunsetothers{⟨multi-label⟩}
\end{verbatim}
### 7.11. Variants of \texttt{mgls}

The commands listed in this section all behave like \texttt{mgls}. These (including \texttt{mgls} itself) are collectively referred to as the \texttt{mgls}-like commands. All commands unset the multi-entry first use flag (unless the \texttt{multiunset}=none option is applied). Only those commands that use the \texttt{gls}-like commands (such as \texttt{gls} or \texttt{glspl}) will unset the individual entry’s first use flag.

#### 7.11.1. \texttt{gls}-like

<table>
<thead>
<tr>
<th>Command</th>
<th>Modifiers</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{glspl[⟨options⟩]{⟨multi-label⟩}⟨insert⟩}}</td>
<td>* + ⟨alt-mod⟩</td>
<td>This uses \texttt{glspl} instead of \texttt{gls} for each entry.</td>
</tr>
<tr>
<td>\texttt{glsmainpl[⟨options⟩]{⟨multi-label⟩}⟨insert⟩}}</td>
<td>* + ⟨alt-mod⟩</td>
<td>This uses \texttt{glspl} instead of \texttt{gls} for the main entry and \texttt{gls} for all the other entries.</td>
</tr>
<tr>
<td>\texttt{Gls[⟨options⟩]{⟨multi-label⟩}⟨insert⟩}}</td>
<td>* + ⟨alt-mod⟩</td>
<td>This uses \texttt{Gls} for the first entry and \texttt{gls} for the other entries.</td>
</tr>
<tr>
<td>\texttt{MGls[⟨options⟩]{⟨multi-label⟩}⟨insert⟩}}</td>
<td>* + ⟨alt-mod⟩</td>
<td>This uses \texttt{MGls} for all entries.</td>
</tr>
<tr>
<td>\texttt{MGlspl[⟨options⟩]{⟨multi-label⟩}⟨insert⟩}}</td>
<td>* + ⟨alt-mod⟩</td>
<td>This uses \texttt{MGlspl} for the first entry and \texttt{glspl} for the remaining entries.</td>
</tr>
</tbody>
</table>

The first entry uses the appropriate sentence case command. The plural form is used for the main entry. So, if the first entry is also the main entry, \texttt{Glspl} is used, otherwise \texttt{Gls} is used. For the remaining entries, \texttt{glspl} will be used if the entry is the main one, otherwise...
7. Multi (or Compound) Entries

\gls will be used.

\MGLsp1[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This uses \Glsp1 for all entries.

\MGLsmainp1[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This uses \Glsp1 for the main entry and \Gl for the other entries.

\MGLS[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This uses \GLS for all entries.

\MGLSp1[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This uses \GLSp1 instead of \Gl for each entry.

\MGLSmainp1[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This uses \GLSp1 for the main entry and \GLS for the others.

7.11.2. Abbreviations

\mglsshort[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This will use \glxstrshort for any entries that have the short field set and will use \glstext otherwise.

\mglslong[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This will use \glxstrlong for any entries that have the long field set and will use \glstext otherwise.

\mglsfull[⟨options⟩]{⟨multi-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
This will use \glsxtrfull for any entries that have the \texttt{short} field set and will use \glsfirst otherwise.

\begin{verbatim}
\Mglsshort[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}

As \texttt{\Mglsshort} but sentence case for the first entry.

\begin{verbatim}
\Mglslong[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}

As \texttt{\Mglslong} but sentence case for the first entry.

\begin{verbatim}
\Mglssfull[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}

As \texttt{\Mglssfull} but sentence case for the first entry.

\subsection{Other Fields}

\begin{verbatim}
\mglsname[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}

This uses \texttt{\mglsname} for each entry.

\begin{verbatim}
\Glsname[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}

This uses \texttt{\Glsname} for the first entry and \texttt{\mglsname} for the remaining entries.

\begin{verbatim}
\MGlsname[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}

This uses \texttt{\MGlsname} for each entry.

\begin{verbatim}
\glssymbol[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}

This uses \texttt{\glssymbol} for each entry if the \texttt{symbol} field has been set, otherwise it uses \texttt{\gls}.

\begin{verbatim}
\MGlssymbol[⟨options⟩]{⟨multi-label⟩}⟨insert⟩
\end{verbatim}
7. Multi (or Compound) Entries

This uses $\texttt{\textbackslash glssymbol}$ if the symbol field has been set otherwise it uses $\texttt{\textbackslash Gls}$ for each element. (Note that no case change is applied to the symbol as this usually isn’t appropriate.)

\begin{verbatim}
\texttt{\Mglssymbol}\{\langle\textit{options}\rangle\}\{\langle\textit{multi-label}\rangle\}\{\langle\textit{insert}\rangle\} \hspace{1cm} \textit{modifiers: * + (alt-mod)}
\end{verbatim}

As $\texttt{\Mglssymbol}$, but $\texttt{\Gls}$ is only used for the first element (if it doesn’t have the symbol field set).

\begin{verbatim}
\texttt{\mglssymbol}\{\langle\textit{options}\rangle\}\{\langle\textit{multi-label}\rangle\}\{\langle\textit{insert}\rangle\} \hspace{1cm} \textit{modifiers: * + (alt-mod)}
\end{verbatim}

If the field given by $\texttt{\mglsfield}$ exists for an element, $\texttt{\glsdisp}$ will be used for that element, with the link text obtained from the field value (followed by the $\langle\textit{insert}\rangle$), otherwise $\texttt{\gls}$ will be used.

\begin{verbatim}
\texttt{\mglsfield} \hspace{1cm} \textit{initial: useri}
\end{verbatim}

This command expands to the internal field label required by $\texttt{\mglsusefield}$. The default value is useri, which corresponds to the user1 key.

\begin{verbatim}
\texttt{\mglsusefield}\{\langle\textit{options}\rangle\}\{\langle\textit{multi-label}\rangle\}\{\langle\textit{insert}\rangle\} \hspace{1cm} \textit{modifiers: * + (alt-mod)}
\end{verbatim}

As $\texttt{\mglsusefield}$ but sentence case for the first element.

\begin{verbatim}
\texttt{\MGlsusefield}\{\langle\textit{options}\rangle\}\{\langle\textit{multi-label}\rangle\}\{\langle\textit{insert}\rangle\} \hspace{1cm} \textit{modifiers: * + (alt-mod)}
\end{verbatim}

As $\texttt{\mglsusefield}$ but sentence case for each element.

You can use the pre-element hook $\texttt{\mglselementprehook}$ to locally redefine $\texttt{\mglsfield}$. Examples:

1. if the multi-category is “sample” then use the user2 field otherwise use the user1 field:

\begin{verbatim}
\renewcommand{\mglselementprehook}{{}\% 
\ifdefstring{\mglscurrentcategory}{sample}{\% 
{\renewcommand{\mglsfield}{userii}}\% 
{\renewcommand{\mglsfield}{useri}}\% 
{\renewcommand{\mglsfield}{useri}}\% }
\end{verbatim}
7. Multi (or Compound) Entries

2. if the element’s category is “sample” then use the user2 field otherwise use the user1 field:

```latex
\renewcommand{\mglselementprehook}{\%
glsifcategory{\mglscurrentlabel}{sample}\
{\renewcommand{\mglfield}{userii}}\
{\renewcommand{\mglfield}{useri}}}%
}
```

3. if either the multi-entry’s category or the element’s category has the custom attribute “\mglfield” set then use it otherwise use the user1 field:

```latex
\renewcommand{\mglselementprehook}{\%
glashascategoryattribute{\mglscurrentcategory}{\mglfield}\%
{\renewcommand{\mglfield}{\glsgetcategoryattribute{\mglscurrentcategory}{\mglfield}}}\
{\%
glashasattribute{\mglscurrentlabel}{\mglfield}\%
{\renewcommand{\mglfield}{\glsgetattribute{\mglscurrentlabel}{\mglfield}}}\
{\renewcommand{\mglfield}{useri}}}%
}
```

7.11.4. Support for glossaries-prefix (\pgls)

If you load the glossaries-prefix package (either after glossaries-extra) or with the prefix package option, then the following commands will use one of the \pgls-like commands provided by that package. (See the glossaries user manual for further details.)

If the glossaries-prefix package hasn’t been loaded then \gls (or analogous case-changing variant) will be used instead and a warning is issued with:

```latex
\mpglsWarning
```

This may be redefined to do nothing to remove the warning.

```latex
\mpgls[(options)]{(multi-label)}{(insert)} modifiers: * + (alt-mod)
```
7. Multi (or Compound) Entries

Uses \texttt{\textbackslash pglspl} for the first element and \texttt{\textbackslash glspl} for the remaining elements.

\begin{verbatim}
\texttt{\textbackslash mpglospl[\langle options\rangle]{\langle multi-label\rangle}\{}{\langle insert\rangle}}
\end{verbatim}

Modifiers: * + \texttt{\langle alt-mod\rangle}

Uses \texttt{\textbackslash pglspl} for the first element and \texttt{\textbackslash glspl} for the remaining elements.

\begin{verbatim}
\texttt{\textbackslash mpglsmainpl[\langle options\rangle]{\langle multi-label\rangle}\{}{\langle insert\rangle}}
\end{verbatim}

Modifiers: * + \texttt{\langle alt-mod\rangle}

Only uses the plural form for the main element. The first element uses the prefixing command (\texttt{\textbackslash pgls} or \texttt{\textbackslash pglspl}, depending on whether the first element is the main element).

\begin{verbatim}
\texttt{\textbackslash Mpgls[\langle options\rangle]{\langle multi-label\rangle}\{}{\langle insert\rangle}}
\end{verbatim}

Modifiers: * + \texttt{\langle alt-mod\rangle}

Uses \texttt{\textbackslash Pgls} for the first element and \texttt{\textbackslash Gls} for the remaining elements.

\begin{verbatim}
\texttt{\textbackslash Mpglsmainpl[\langle options\rangle]{\langle multi-label\rangle}\{}{\langle insert\rangle}}
\end{verbatim}

Modifiers: * + \texttt{\langle alt-mod\rangle}

Only uses the plural form for the main element. The first element uses the prefixing command (\texttt{\textbackslash Pgls} or \texttt{\textbackslash Pglspl}, depending on whether the first element is the main element).

\begin{verbatim}
\texttt{\textbackslash MPGls[\langle options\rangle]{\langle multi-label\rangle}\{}{\langle insert\rangle}}
\end{verbatim}

Modifiers: * + \texttt{\langle alt-mod\rangle}

Uses \texttt{\textbackslash Pgls} for the first element and \texttt{\textbackslash Gls} for the remaining elements.

\begin{verbatim}
\texttt{\textbackslash MPGlspl[\langle options\rangle]{\langle multi-label\rangle}\{}{\langle insert\rangle}}
\end{verbatim}

Modifiers: * + \texttt{\langle alt-mod\rangle}

Uses \texttt{\textbackslash Pglspl} for the first element and \texttt{\textbackslash Glspl} for the remaining elements.

\begin{verbatim}
\texttt{\textbackslash MPGlsmainpl[\langle options\rangle]{\langle multi-label\rangle}\{}{\langle insert\rangle}}
\end{verbatim}

Modifiers: * + \texttt{\langle alt-mod\rangle}

Only uses the plural form for the main element. The first element uses the prefixing command (\texttt{\textbackslash Pgls} or \texttt{\textbackslash Pglspl}, depending on whether the first element is the main element). All
7. Multi (or Compound) Entries

(elements are use sentence case.

\texttt{\textbackslash MPGLS[\langle options\rangle]\{\langle multi-label\}\}[\langle insert\}\]} \textit{modifiers: * + \langle alt-mod\rangle}

Uses \texttt{\textbackslash PGLS} for the first element and \texttt{\textbackslash GLS} for the remaining elements.

\texttt{\textbackslash MPGLSp1[\langle options\rangle]\{\langle multi-label\}\}[\langle insert\}\]} \textit{modifiers: * + \langle alt-mod\rangle}

Uses \texttt{\textbackslash PGLSp1} for the first element and \texttt{\textbackslash GLSp1} for the remaining elements.

\texttt{\textbackslash MPGLSmainpl[\langle options\rangle]\{\langle multi-label\}\}[\langle insert\}\]} \textit{modifiers: * + \langle alt-mod\rangle}

Only uses the plural form for the main element. All elements are converted to all caps. The first element uses the prefixing command (\texttt{\textbackslash PGLS} or \texttt{\textbackslash PGLSp1}, depending on whether the first element is the main element).

7.12. Cross-References

Multi-entry labels may be used in the cross-referencing keys \texttt{see} and \texttt{sealso}. The formatting command will use:

\texttt{\selectlanguage{en}\textbackslash mglsseefirstitem\{\langle multi-label\}\}} for the first item in the list (if it's a multi-entry) and

\texttt{\selectlanguage{en}\textbackslash mglsseeitem\{\langle multi-label\}\}} for any subsequent items that are multi-entries. The default definition of \texttt{\textbackslash mglsseeitem} is:

\begin{verbatim}
\newcommand*{\mglsseeitem}[1]{\/mglsname[all=noindex, setup={hyper=allmain}]{#1}}
\end{verbatim}

This switches off indexing, sets the hyperlink to encompass the entire multi-entry content and uses the name field. The default definition of \texttt{\textbackslash mglsseefirstitem} is simply \texttt{\textbackslash mglsseeitem}.

For example, to use the \texttt{short} or \texttt{text} fields:
7. Multi (or Compound) Entries

A multi-entry label may also be used in the `alias` key. The hyperlink target will be the target for the main entry. For example:

```
\multiglossaryentry{cbot}{clostridium,botulinum}
\newglossaryentry{botox}{name={botox},description={},alias={cbot}}
```

In this case \gls{botox} will hyperlink to the botulinum target.

Any multi-entries used in the cross-referencing keys must be defined before the glossary is displayed. There is some support for `docdef=true` but not for the other `docdef` settings.

7.13. Additional Commands

You can test if a label represents a multi-entry using:

```
\glsxtrifmulti{⟨multi-label⟩}{⟨true⟩}{⟨false⟩}
```

This does ⟨true⟩ if a multi-entry has been defined with the label ⟨multi-label⟩ otherwise it does ⟨false⟩.

```
\glsxtrmultimain{⟨multi-label⟩}
```

Expands to the main entry label for the multi-entry identified by ⟨multi-label⟩ or nothing if not defined.

```
\glsxtrmultilist{⟨multi-label⟩}
```

Expands to the list of element labels for the multi-entry identified by ⟨multi-label⟩ or nothing if not defined.
7. Multi (or Compound) Entries

Iterates over all the list of element labels for the multi-entry identified by \textlangle multi-label\rangle. This defines \textlangle cs\rangle to the current element label on each iteration of the loop, which can be used to reference the label in \textlangle body\rangle. This internally uses \textbackslash@for (patched by the xfor package, which allows the loop to be broken).

\begin{verbatim}
\textbackslash mglsforotherelements\lbrace\textlangle multi-label\rangle\rbrace\{\textlangle cs\rangle\}\{\textlangle body\rangle\}
\end{verbatim}

As \textbackslash mglsforelements but skips the main entry label.

\begin{verbatim}
\textbackslash glsxtrmultitotalelements\lbrace\textlangle multi-label\rangle\rbrace
\end{verbatim}

Expands to the total number of elements in the given multi-entry or nothing if \textlangle multi-label\rangle hasn’t been defined.

\begin{verbatim}
\textbackslash glsxtrmultimainindex\lbrace\textlangle multi-label\rangle\rbrace
\end{verbatim}

Expands to the index of the main element in the given multi-entry or nothing if \textlangle multi-label\rangle hasn’t been defined. Indexing starts from 1 for the first element.

\begin{verbatim}
\textbackslash glsxtrmultilastotherindex\lbrace\textlangle multi-label\rangle\rbrace
\end{verbatim}

Expands to the index of the final non-main element in the given multi-entry or nothing if \textlangle multi-label\rangle hasn’t been defined.

The \textbackslash multiglossaryentry command will write the label information to the aux file using:

\begin{verbatim}
\textbackslash writemultiglossentry\lbrace\textlangle options\rangle\rbrace\lbrace\textlangle multi-label\rangle\rbrace\lbrace\textlangle main-label\rangle\rbrace\lbrace\textlangle list\rangle\rbrace
\end{verbatim}

This is will write the following line to the aux file:

\begin{verbatim}
\textbackslash@ glsxtrmultientry\lbrace\textlangle options\rangle\rbrace\lbrace\textlangle multi-label\rangle\rbrace\lbrace\textlangle main-label\rangle\rbrace\lbrace\textlangle list\rangle\rbrace
\end{verbatim}

This is provided to support \texttt{docdef=true} and also for the benefit of any tools that require the information (such as \texttt{bib2gls} or autocompletion tools). If it’s not required and causes too much clutter, it can be disabled by redefining \textbackslash writemultiglossentry to do nothing.
7. Multi (or Compound) Entries

### 7.14. bib2gls

In the bib2gls v2.9+ user manual, these multi-entry sets are referred to as “compound entries” or “compound sets” to differentiate them from bib2gls’s multi-entry types (such as @dualentry).

Each instance of one of the \mglslike commands is written to the aux file and so can be picked up by bib2gls (at least version 2.9). The resource option can be used to determine whether or not to consider the other (non-main) elements to be dependent on the main element.

With bib2gls, you can either define the compound entries in the document with \multiglossaryentry (or \providemultiglossaryentry) or you can use the @compoundset entry type in a bib file. Whichever method you use, remember that the entries that form the elements of the set must be defined first. See the bib2gls manual (v2.9+) for further details.

You can use the resource option compound-adjust-name to replace the name field of the main entry to:

\[\text{\glsxtrmultientryadjustedname}{\langle\text{sublist1}\rangle}\{\langle\text{name}\rangle}\{\langle\text{sublist2}\rangle}\{\langle\text{multi-label}\rangle}\]

where \langle multi-label \rangle is the label identifying the compound set, \langle name \rangle was the value of the name before adjustment, \langle sublist1 \rangle is the sub-list of other element labels before the main element (empty if the main element is the first element in the list), and \langle sublist2 \rangle is the sub-list of other elements after the main element (empty if the main label is the final element).

This command is defined in glossaries-extra-bib2gls, which is automatically loaded with record=only and record=nameref. Case-changing versions of this command are also available.

\[\text{\Glsxtrmultientryadjustedname}{\langle\text{sublist1}\rangle}\{\langle\text{name}\rangle}\{\langle\text{sublist2}\rangle}\{\langle\text{multi-label}\rangle}\]

This is a sentence case version of \glsxtrmultientryadjustedname.

\[\text{\GlsXtrmultientryadjustedname}{\langle\text{sublist1}\rangle}\{\langle\text{name}\rangle}\{\langle\text{sublist2}\rangle}\{\langle\text{multi-label}\rangle}\]

This is a title case version of \glsxtrmultientryadjustedname.

\[\text{\GLSxtrmultientryadjustedname}{\langle\text{sublist1}\rangle}\{\langle\text{name}\rangle}\{\langle\text{sublist2}\rangle}\{\langle\text{multi-label}\rangle}\]

372
7. Multi (or Compound) Entries

This is an all caps version of \texttt{\glsxtrmultientryadjustedname}.

Note that the above commands don’t take the prefix or suffix into account (see §7.3).

The separator between each element in the sub-lists is produced with:

\begin{verbatim}
\glsxtrmultientryadjustednamesep\{⟨pre-label⟩\}\{⟨post-label⟩\}
\end{verbatim}

The default definition just uses \texttt{\gscombinedfirstsepfirst}.

The separator between the last element of \texttt{⟨sublist1⟩} and the main element is produced with:

\begin{verbatim}
\glsxtrmultientryadjustednamepresep\{⟨pre-label⟩\}\{⟨post-label⟩\}
\end{verbatim}

Similarly for the separator between the main element and the first element of \texttt{⟨sublist2⟩}:

\begin{verbatim}
\glsxtrmultientryadjustednamepostsep\{⟨pre-label⟩\}\{⟨post-label⟩\}
\end{verbatim}

These both default to \texttt{\glsxtrmultientryadjustednamesep}.

The \texttt{⟨name⟩} is encapsulated with:

\begin{verbatim}
\glsxtrmultientryadjustednamefmt\{⟨text⟩\}
\end{verbatim}

This just does its argument by default. If \texttt{⟨sublist1⟩} is empty for the sentence case version, then \texttt{⟨name⟩} is encapsulated with:

\begin{verbatim}
\GlsXtrmultientryadjustednamefmt\{⟨text⟩\}
\end{verbatim}

This does \texttt{\makefirstuc{⟨text⟩}} by default. For the title case version, the name is encapsulated with:

\begin{verbatim}
\GlsXtrmultientryadjustednamefmt\{⟨text⟩\}
\end{verbatim}

This uses \texttt{\glscapitalisewords}, if defined, or \texttt{\capitalisewords} otherwise. The all caps version uses:

\begin{verbatim}
\GLSxtrmultientryadjustednamefmt\{⟨text⟩\}
\end{verbatim}

This uses \texttt{\mfirstucMakeUppercase} by default.
7. Multi (or Compound) Entries

Each element label in the sub-lists is encapsulated with:

\glsxtrmultientryadjustednameother{⟨text⟩}

This does \glsentryname{⟨label⟩} by default. For the sentence case version (where ⟨sub-list1⟩ isn’t empty), then the element label is encapsulated with:

\Glsxtrmultientryadjustednameother{⟨text⟩}

This does \Glsentryname{⟨label⟩} by default. The title case version uses:

\GlsXtrmultientryadjustednameother{⟨text⟩}

This does \glsentrytitlecase{⟨label⟩}{name} by default. The all caps version uses:

\GLSxtrmultientryadjustednameother{⟨text⟩}

This is defined as:

\newcommand*{\GLSxtrmultientryadjustednameother}{[1]{%  \mfirstucMakeUppercase{\glsentryname{#1}}}}
8. Defining and Displaying Glossaries

As with the base glossaries package, you need to establish the indexing method in the pream-
ble and use the appropriate \print⟨...⟩glossary command. For example, to use the “noidx”
method you need \makenoidxglossaries in the preamble and \printnoidxglossary in
the document. Whereas if you want to use makeindex or xindy, you need \makeglossaries
in the preamble and \printglossary in the document. The glossaries-extra package pro-
vides a hybrid approach:

\makeglossaries[⟨types⟩]

If the optional argument is present, then ⟨types⟩ should be a comma-separated list of glossary
labels that should be processed by makeindex or xindy as per the normal behaviour of
\makeglossaries. Any non-ignored glossaries that are not listed in ⟨types⟩ should be
treated as though \makenoidxglossaries was used. This means that the glossaries listed in
⟨types⟩ should be displayed using \printglossary and the other (non-ignored glossaries)
should be displayed with \printnoidxglossary. See sample-mixedsort.tex for an ex-
ample.

If the optional argument is omitted, it will be treated as per the original \makeglossaries
provided by the base glossaries package. If the optional argument is present, \glsindexing-
setting will be set to makeindex-noidx or xindy-noidx, depending on whether make-
index or xindy should be used.

Glossaries can’t be defined after \makeglossaries. This ensures that all the required
indexing files are opened. If you’re not using \makeglossaries, glossaries need to
be defined before any entries that should belong to them are defined.

The base glossaries package provides \newignoredglossary to define an ignored glossary
that doesn’t have any associated indexing files. This will automatically switch off hyperlinks
for any entries assigned to the glossary (since there will be no target). With glossaries-extra,
it’s possible to have targets without using the indexing methods provided by the base pack-
age. For example, it’s possible to have standalone entries (see §8.5) or targets can be created
with \printunsrtglossary, so glossaries-extra provides a starred version:

\newignoredglossary*{⟨glossary-label⟩}
This behaves like the unstarred version but doesn’t disable hyperlinks. The glossary will still be omitted by iterative commands, such as `\printunsrtglossaries`, and can’t be used with `\printglossary` or `\printnoidxglossary`. If you use an ignored glossary with `\printunsrtglossary`, you will need to use the `title` option to override the default title, if required.

\provideignoredglossary\{⟨glossary-label⟩\}

This is like `\newignoredglossary` but does nothing if the glossary has already been defined.

With the indexing options provided by the base glossaries package, if you want a term to appear in more than one glossary, it’s necessary to define a duplicate entry with a different label. With the “unsrt” family of commands, the same entry can appear in multiple glossaries. This can be done by simply copying the entry’s label to the required glossary using:

\glsxtrcopytoglossary\{⟨entry-label⟩\}\{⟨glossary-type⟩\}

This just adds the label to the target glossary’s internal comma-separated list that commands like `\printunsrtglossary` iterate over. The unstarred version locally adds the label. The starred version performs a global change.

\glsxtrcopytoglossary is not compatible with `docdef=true`.

Note that the `type` field will still be set to the original glossary. This is considered the entry’s primary glossary. There’s no field that keeps track of the additional glossaries the entry has been copied to.

If used with `\printglossary` or `\printnoidxglossary`, the entry will only be indexed for its primary glossary. It won’t show up in the other glossaries, but will be found when using an iterative command, such as `\glsaddall`, over the target glossary.

The base glossaries package provides `\forallglossaries` to iterate over a list of glossaries labels (all non-ignored glossaries by default). This can also be used with `glossaries-extra` but `\forallacronyms` is only for glossaries that have been declared as lists of acronyms, so it’s inappropriate with the `glossaries-extra` package. Instead, you can use the analogous command:

\forallabbreviationlists\{⟨cs⟩\}\{⟨body⟩\}

Each instance of `\newabbreviation` will add the abbreviation’s associated glossary (identified by the `type` key) to the internal list of labels (if not already added). Note that this won’t take into account any glossaries that had abbreviations copied or moved to it.
8. Defining and Displaying Glossaries

8.1. Entry Page Reference

The base glossaries package provides \glsrefentry{⟨entry-label⟩}, which uses \ref to reference the entry’s associated counter (enabled with entrycounter or subentrycounter, not the location counter). The glossaries–extra package additionally provides:

\glsxtrpagerref{⟨entry-label⟩}

This works in the same way but uses \pageref instead of \ref. As with \glsrefentry, if the corresponding counter has not been enabled, this just does \gls{⟨entry-label⟩}.

8.2. Glossary Preamble

The base package provides \glossarypreamble, which is used at the start of the glossary. By default, this will use the preamble associated with the current glossary. If you redefine \glossarypreamble, this will set the preamble for all glossaries. To set the preamble for a particular glossary, you can use \setglossarypreamble. With glossaries–extra, you can additionally append to an existing preamble using:

\apptoglossarypreamble[⟨type⟩]{⟨text⟩}

This (locally) appends ⟨text⟩ to the preamble for the glossary identified by ⟨type⟩. If ⟨type⟩ is omitted, \glsdefaulttype is assumed.

\pretoglossarypreamble[⟨type⟩]{⟨text⟩}

This (locally) prepends ⟨text⟩ to the preamble for the glossary identified by ⟨type⟩. If ⟨type⟩ is omitted, \glsdefaulttype is assumed.

8.3. Options

In addition to the options available with \printglossary, the following options are also provided. Some of these listed here are specific to \printunsrtglossary and printunsrtglossarywrap. Options provided by the base package that aren’t available for the “unsrt” family of commands are identified below.

\setlength{⟨option⟩}{⟨value⟩}

377
8. Defining and Displaying Glossaries

This option is only available for \printnoidxglossary. The \printunsrtglossary and \printunsrtinnerglossary commands simply iterate over the glossary’s internal list in the order in which the entries have been added to that glossary. If you are using bib2gls, use the sort resource option instead.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title=⟨text⟩</td>
<td>This option is provided by the base glossaries package to override the default title for the glossary. This option is also available for \printunsrtglossary and printunsrtglossarywrap but not for \printunsrtinnerglossary.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>toctitle=⟨text⟩</td>
<td>This option is provided by the base glossaries package to override the default table of contents title for the glossary. This option is also available for \printunsrtglossary and printunsrtglossarywrap but not for \printunsrtinnerglossary.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numberedsection=⟨value⟩</td>
<td>This option is provided by the base glossaries package to indicate whether or not the section header used at the start of the glossary should be numbered rather than unnumbered (and whether or not to automatically label the glossary with \label{\glsautoprefix(glossary-type)}). The numberedsection package option will change the default setting to match. This option is not available for \printunsrtinnerglossary.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>style=⟨style-name⟩</td>
<td>This option is provided by the base glossaries package to set the glossary style. This option is not available for \printunsrtinnerglossary.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label=⟨label⟩</td>
<td>This option is provided by glossaries-extra to add \label{⟨label⟩} after the section header. This is essentially like numberedsection=nameref but you supply the label. This option is not available for \printunsrtinnerglossary. Alternatively, you can use:</td>
</tr>
</tbody>
</table>

\glsxtrsetglossarylabel{⟨label⟩}
8. Defining and Displaying Glossaries

This will need to be scoped or changed between glossaries or use a command in \(\langle \text{label} \rangle\) that expands differently for each glossary to avoid duplicate labels.

\[
\text{leveloffset}=\langle \text{offset} \rangle
\]

This option sets or increments the hierarchical level offset. If \(\langle \text{offset} \rangle\) starts with ++ then the current offset is incremented by the given amount otherwise the current offset is set to \(\langle \text{offset} \rangle\). For example, an entry with a normal hierarchical level of 1 will be treated as though it has hierarchical level \(1 + \langle \text{offset} \rangle\). Note that the glossary style may not support the resulting hierarchical level. This option is only available for the “unsrt” family of commands and the \texttt{printunsrtglossarywrap} environment. See §8.4.3.1 for an example.

\[
\text{flatten}=\langle \text{boolean} \rangle
default: \text{true}; \ initial: \text{false}
\]

Treats all entries as though they have the same hierarchical level (the value of leveloffset). This option is only available for the “unsrt” family of commands and the \texttt{printunsrtglossarywrap} environment. Unlike the flatten resource option, this option doesn’t actually remove the parent field.

\[
\text{groups}=\langle \text{boolean} \rangle
default: \text{true}; \ initial: \text{true}
\]

This option is only applicable to the “unsrt” family of commands and \texttt{printunsrtglossarywrap}. If set to false, it will prevent groups from being formed. If true (the default), groups will only be formed if they are supported. See §8.4 for further details.

\[
\text{prefix}=\langle \text{prefix} \rangle
\]

This option is provided by glossaries-extra and simply redefines \texttt{\glolinkprefix} to expand to \(\langle \text{prefix} \rangle\). If hyperlinks are supported and the glossary style uses \texttt{\glstarget} to create the entry’s hypertarget, the target name is obtained from \texttt{\glolinkprefix(\text{entry-label})}. If you are displaying multiple glossaries with shared entries (for example, using the secondary resource option with \texttt{bib2gls}), then changing the prefix can avoid duplicate targets.

Note that this option will also affect the targets used by the \texttt{\gls}-like and \texttt{\glstext}-like commands. This means that if you have, for example, \texttt{\gls} in the description of an entry, then its hyperlink will go to that entry’s item in the current glossary. Whereas referencing that entry outside of the glossary will hyperlink to the glossary that uses the prefix matching the setting at that point in the document. For example:
This uses the \texttt{showtargets} package option to show the target names to the left of the hyperlink or hypertarget. The result is:

Example 132: Changing the target prefix

Link to glossary 1: \texttt{[glo:sample]sample}.  
Link to glossary 2: \texttt{[other-sample]sample}.

\textbf{Glossary}

\texttt{[glo:sample]sample}: an example description that references \texttt{[glo:another]another}.

\texttt{[glo:another]another}: some other example description that references \texttt{[glo:sample]sample}.

\textbf{Glossary}

\texttt{[other-sample]sample}: an example description that references \texttt{[other-another]another}.

\texttt{[other-another]another}: some other example description that references \texttt{[other-sample]sample}.

Within the main part of the document, the first reference to “sample” has a hyperlink to the
first glossary (with the target \texttt{glo:sample}, which uses the default prefix), and the second reference has a hyperlink to the second glossary (with the target \texttt{other-sample}).

Within the glossaries, the \texttt{\gls} references use the current glossary prefix, so the target is in the same glossary.

\begin{Verbatim}
\texttt{\targetnameprefix=\langle\texttt{prefix}\rangle}
\end{Verbatim}

This is similar to the previous option but only affects the prefix for the entry item's target and doesn't change the prefix for any references contained within the glossary. This \textit{prepends} the given prefix to the default prefix.

If the above example is modified to:

\begin{Verbatim}
\texttt{Link to glossary 1: \gls{sample}}.
\texttt{Link to glossary 2: \gls[prefix=other-glo:]{sample}}.
\end{Verbatim}

\begin{Verbatim}
\texttt{\textbackslash printunsrtglossary}
\texttt{\textbackslash printunsrtglossary[targetnameprefix=other-]}
\end{Verbatim}

Then the result will be:

\begin{Verbatim}
Example 133: Prepending to the target prefix for just the entry item
\begin{itemize}
\item Link to glossary 1: \texttt{[glo:sample]\_\texttt{sample}}.
\item Link to glossary 2: \texttt{[other-glo:sample]\_\texttt{sample}}.
\end{itemize}
\end{Verbatim}

\begin{Verbatim}
Glossary
\texttt{[glo:sample]\_\texttt{sample}} an example description that references \texttt{[glo:another}\_\texttt{another}}.
\texttt{[glo:another]\_\texttt{another}} some other example description that references \texttt{[glo:sample]\_\texttt{sample}}.
\end{Verbatim}

\begin{Verbatim}
Glossary
\texttt{[other-glo:sample]\_\texttt{sample}} an example description that references \texttt{[glo:another}\_\texttt{another}}.
\texttt{[other-glo:another]\_\texttt{another}} some other example description that references \texttt{[glo:sample]\_\texttt{sample}}.
\end{Verbatim}
Note that this has prepended other- to the existing gloc: prefix. This is why the prefix option in the second \gls reference had to be changed to match the appropriate hypertarget name. The \gls references in the second glossary now point to the relevant line in the first glossary.

It’s possible to combine targetnameprefix with prefix={} but that will also affect the \gls references within the glossary.

```
target=⟨boolean⟩  default: true; initial: true
```

This is a boolean option that can be used to switch off the automatic creation of the entry hypertargets but still allows hyperlinks within the glossary. This can be used to prevent duplicate destinations for secondary glossaries.

```
preamble=⟨text⟩
```

Redefines \glossarypreamble to ⟨text⟩.

```
postamble=⟨text⟩
```

Redefines \glossarypostamble to ⟨text⟩.

---

### 8.4. Displaying a Glossary Without Sorting or Indexing

The base glossaries package provides two ways of displaying a glossary, depending on the indexing option: \printglossary (used with \makeglossaries) or \printnoidxglossary (used with \makenoidxglossaries).

The glossaries-extra package provides an alternative that doesn’t require sorting or indexing.

```
\printunsrtglossary[⟨options⟩]
```

This behaves in a similar way to \printnoidxglossary, but it always lists all the defined entries for the given glossary in the order in which they were added to the glossary. Unlike \printglossary, you may use \printunsrtglossary with an ignored glossary.

The “unsrt” family of commands and printunsrtglossarywrap are not intended for use with \makeglossaries and \makenoidxglossaries. Mixing these different methods can result in unexpected behaviour.
There is also a starred version which has a mandatory argument:

\printunsrtglossary*[/⟨options⟩]{⟨init-code⟩}

This is equivalent to:

\begingroup
  ⟨init-code⟩\printunsrtglossary{/⟨options⟩}
\endgroup

There’s no significant difference between doing:

{⟨init-code⟩}\printunsrtglossary[/⟨options⟩]

and

\printunsrtglossary*[/⟨options⟩]{⟨init-code⟩}

Note that unlike \glossarypreamble, the supplied ⟨init-code⟩ is done before the glossary header.

If you want to use one of the tabular-like styles with \printunsrtglossary, make sure you load glossaries-extra-stylemods which modifies the definition of \glsautotoc to avoid the “Incomplete \texttt{\iftrue}” error that may otherwise occur.

As with \printglossary and \printnoidxglossary, there is also a command to print all non-ignored glossaries in the order in which they were defined:

\printunsrtglossaries

This means you now have the option to simply list all entries on the first \LaTeX run without the need for a post-processor, however there will be no location list in this case, as that has to be set by a post-processor such as bib2gls (see §11).

No attempt is made to gather hierarchical elements. If child entries aren’t defined immediately after their parent entry, they won’t be together in the glossary when using \printunsrtglossary.

The way that \printunsrtglossary basically works is to iterate over every label in the glossary’s internal label list and format each entry according to the way the glossary style
would normally format the entry’s hierarchical level (described in more detail in §8.4.3). If a change in letter group is detected, the letter group heading and group skip will be inserted.

A label is appended to the glossary’s internal label list whenever an entry is defined. This means that the list will normally be in order of definition, but it’s also possible to copy an entry’s label to another glossary’s internal label list using \glsxtrcopytoglossary, which can be used to provide a different order. A simple example:

```latex
\documentclass{article}
\usepackage[style=treegroup]{glossaries-extra}
\newglossaryentry{ant}{name={ant},description={}}
\newglossaryentry{gazelle}{name={gazelle},description={}}
\begin{document}
\printunsrtglossary
\end{document}
```

The document build only requires one \TeX call in this case.

Example 134: Displaying unsorted glossaries

Glossary

A
ant
G
gazelle

Note the difference if the stylemods option is used:

```latex
\usepackage[stylemods,style=treegroup]{glossaries-extra}
```
Example 135: Displaying unsorted glossaries with \texttt{stylemods}

\begin{itemize}
\item \texttt{Glossary}
\item 65
\item ant
\item 71
\item gazelle
\end{itemize}

In this case, the group headings are now numbers instead of letters. The styles provided with \texttt{glossaries-extra} and those modified by \texttt{glossaries-extra-stylemods} are designed to assist integration with \texttt{bib2gls}. Without these modifications, \texttt{\printnoidxglossary} behaves like the less sophisticated \texttt{\printnoidxglossary} which checks if the label is an integer less than 256 and uses \texttt{\char} to create the title (if no title has been provided).

If you really want to use \texttt{\printunsrtglossary} without \texttt{bib2gls} and you want letter groups with \texttt{stylemods} without having to define all the titles, you can use:

\begin{itemize}
\item \texttt{\glsxtrnoidxgroups}
\end{itemize}

which will switch over to using the group titling method used with \texttt{\printnoidxglossary} (which only supports ASCII). This command is only available with \texttt{record=off} and can’t be used with \texttt{\makeglossaries}.

If, conversely, you don’t want any groups formed, regardless of the glossary style, you can disable them with \texttt{groups=false}. The above example can be modified so that the document contains:

\begin{itemize}
\item \texttt{\printunsrtglossary[title=\{Glossary 1\}]}
\item \texttt{\glsxtrnoidxgroups}
\item \texttt{\printunsrtglossary[title=\{Glossary 2\}]}
\item \texttt{\printunsrtglossary[groups=false,title=\{Glossary 3\}]}
\item \texttt{\printunsrtglossary[style=tree,nogroupskip,title=\{Glossary 4\}]}
\end{itemize}

This repeats the same glossary. The first is the same as the previous example. The second is the same as the example that didn’t use \texttt{stylemods}. The final two glossaries have the groups suppressed. Using \texttt{groups=false} (Glossary 3) is more efficient than using \texttt{nogroupskip} and switching to a style that doesn’t show the header (Glossary 4).

I’ve also switched to two column mode to display the result in a more compact form. The first two glossaries are shown on the left and the last two are on the right:
### Example 136: Displaying unsorted glossaries with different group settings

<table>
<thead>
<tr>
<th>Glossary 1</th>
<th>Glossary 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>65 ant</td>
<td>ant gazelle</td>
</tr>
<tr>
<td>71 gazelle</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Glossary 2</th>
<th>Glossary 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>A ant G gazelle</td>
<td>ant gazelle</td>
</tr>
</tbody>
</table>

The “unsrt” family of commands were designed for use with bib2gls, which uses more complex alphanumeric group labels to allow for greater customization and to avoid conflict where there are multiple glossaries or hierarchical levels with potentially the same letter groups.

The way that bib2gls works is to select entries from a bib file, according to the document requirements, sort the entries, and then write the entry definitions (with commands like \longnewglossaryentry* or \newabbreviation) in the glstex in the desired order, which is then input by \GlsXtrLoadResources. This means that \printunsrtglossary will display the entries in that order since, from glossaries-extra’s point of view, that’s the order of definition.

While it is possible to use \printunsrtglossary without bib2gls, as in the above example, for long or complex glossaries it’s better to use bib2gls which can automatically assign appropriate titles to the groups.

Groups and hierarchy are discussed in more detail in §8.4.1. See §8.4.2 for location lists and §8.4.3.1. Advanced commands and further detail about the way \printunsrtglossary works are covered in §8.4.3.
8. Defining and Displaying Glossaries

8.4.1. Groups and Hierarchy

See Gallery: Logical Glossary Divisions (type vs group vs parent)\textsuperscript{a} for the difference between the group, type and parent fields.

\textsuperscript{a}dickimaw-books.com/gallery/index.php?label=logicaldivisions

Here’s a longer example that uses \texttt{stylemods} to automatically load glossary-bookindex:

\begin{verbatim}
\documentclass{article}
\usepackage[stylemods=bookindex,style=bookindex]{glossaries-extra}
\newglossaryentry{waterfowl}{name={waterfowl},description={}}
\newglossaryentry{ant}{name={ant},description={}}
\newglossaryentry{adder}{name={adder},description={}}
\newglossaryentry{duck}{name={duck},parent={waterfowl},description={}}
\newglossaryentry{zebra}{name={zebra},description={}}
\newglossaryentry{aardvark}{name={aardvark},description={}}
\newglossaryentry{gazelle}{name={gazelle},description={}}
\newglossaryentry{mallard}{name={mallard},parent={duck},description={}}
\newglossary*{another}{Another Glossary}
\glsxtrcopytoglossary{mallard}{another}
\glsxtrcopytoglossary{aardvark}{another}
\glsxtrcopytoglossary{zebra}{another}
\glsxtrcopytoglossary{ant}{another}
\glsxtrcopytoglossary{duck}{another}
\begin{document}
\printunsrtglossary
\printunsrtglossary[type=another]
\end{document}
\end{verbatim}

Unlike the previous examples that defined the entries in alphabetical order, this example hasn’t used any logical order. Note, in particular, that the child entries “duck” and “mallard” (which have the parent key set) have not been defined immediately after their parent.

The first \texttt{\printunsrtglossary} has the default \texttt{type=main} and lists all entries defined in the main glossary, in the order in which they were defined. The second \texttt{\printunsrtglossary[type=another]} lists all entries in the custom another glossary and is in the order in which the entries were copied to that glossary.

The document build again simply requires one \LaTeX call. The result is shown in Example 137 on the following page.
Example 137: Displaying unsorted glossaries with a copied list

Glossary

87  65
waterfowl

65  aardvark
ant
adder
duck

90  gazelle
zebra
mallard

Another Glossary

mallard  65

65  ant
aardvark
duck

90
zebra

There are some oddities in both lists. It’s the glossary style that determines the formatting of the entries according to the entry’s hierarchical level, but it looks strange for the duck and mallard entries to be indented when they don’t follow after their parent entry.

As the internal loop within \printunsrtglossary iterates over each entry, it tries to determine which letter group the entry belongs to. If it’s different from the group for the previous entry (in the same hierarchical level), a group header is added (which may or may not be displayed, depending on the glossary style). This means than an unordered list of entries, such as in the above example, may contain repeated headers.

The way that the group is determined depends on whether or not the group key has been defined. If it isn’t defined (the default), then the group label is obtained from the uppercase character code of the first token of the sort key. If the token doesn’t have an uppercase character code (indicating that it’s not a letter) or if the sort value is empty then the label will be set to glssymbols (which corresponds to the symbol group). This is the same way that \printnoidxglossary inserts groups.
Remember that if the sort key hasn’t been set, it will be assigned automatically to the same value as the name key (or with sort=use or sort=def to a numerical value). The sort key will be empty if you use sort=clear. The sort=none setting simply skips the pre-processing of the sort key (such as sanitizing).

For example, the ant entry doesn’t explicitly use the sort key, so the sort value is obtained from the name key, which is set to ant. The first token is “a”, which is a letter. The group’s label is obtained from the letter’s uppercase character decimal code (65). There’s no associated title (which can be assigned with \glsxtrsetgrouptitle), so the title is simply “65” (with stylemods, see earlier) or “A” (without stylemods or with \glsxtrnoidxgroups).

The ant entry is followed by “adder”. The same process determines that the “adder” group label is also 65. There’s no change in the group label from the previous entry (ant) so no header is inserted.

By default, this group check is omitted for child entries, which is why no group header is inserted before duck or mallard. So the next entry to be checked for a group is the zebra entry, which has the group label 90 (the decimal code for “Z”). Again there’s no title associated with that label so the title is simply the label.

The zebra entry is followed by aardvark which, following the same process, has the group label 65. This is different from the previous group label (90) so a group header is inserted. This is why there are two “90” letter groups.

If the group key has been defined (which is the case with record=only and record=nameref) then the group label is obtained from the group field. If the group field is defined but empty then the entry will belong to the empty group. The value of the sort field is now irrelevant.

So, simply adding the record option to the above example document will cause the group headers to disappear. This is because the group key will now be defined but is empty for each entry. Even with a style like bookindex, there won’t be any group headers.

Provided the group key has been defined, the field used to store the group label is given by:

\glsxtrgroupfield initial: group

This expands to group, by default. However it’s possible to use a different field in which to store the group label, in which case \glsxtrgroupfield will need to be redefined. For example:

{\renewcommand{\glsxtrgroupfield}{othergroup}\printunsrtglossary}

or

389
8. Defining and Displaying Glossaries

\printunsrtglossary*{\renewcommand{\glsxtrgroupfield}{othergroup}}

(but this still requires the \texttt{group} key to be defined, even if it’s not being used to store the group label). With \texttt{bib2gls}, the \texttt{secondary} resource option (combined with \texttt{--group}) will store the group label obtained from the secondary sort in the \texttt{secondarygroup} field and adds the redefinition of \texttt{\glsxtrgroupfield} to the associated glossary preamble. This prevents it from clashing with the \texttt{group} field in the event that the secondary sort method has produced a different set of groups (which is likely).

The follow example document uses \texttt{record} to create the \texttt{group} key and assigns group labels with associated titles. Note that the \texttt{secondarygroup} field doesn’t have an associated key, so it needs to be set with a field assignment command, such as \texttt{\GlsXtrSetField}.

\documentclass{article}
\usepackage[record,stylemods=bookindex,style=bookindex]{glossaries-extra}
\glsxtrsetgrouptitle{group1label}{Group 1}
\glsxtrsetgrouptitle{group2label}{Group 2}
\glsxtrsetgrouptitle{group3label}{Group 3}
\glsxtrsetgrouptitle{group4label}{Group 4}
\newglossaryentry{waterfowl}{name={waterfowl},description={},group={group1label}}
\newglossaryentry{ant}{name={ant},description={},group={group1label}}
\GlsXtrSetField{ant}{secondarygroup}{group4label}
\newglossaryentry{adder}{name={adder},description={},group={group2label}}
\newglossaryentry{duck}{name={duck},parent={waterfowl},description={},group={group4label}}
\GlsXtrSetField{duck}{secondarygroup}{group2label}
\newglossaryentry{zebra}{name={zebra},description={},group={group2label}}
\GlsXtrSetField{zebra}{secondarygroup}{group3label}
\newglossaryentry{aardvark}{name={aardvark},description={},group={group2label}}
\GlsXtrSetField{aardvark}{secondarygroup}{group1label}
\newglossaryentry{gazelle}{name={gazelle},description={},group={group1label}}

390
This is essentially mimicking the way that the `secondary` resource option sets the `secondarygroup` field and adds the redefinition of `\glsxtrgroupfield` to the secondary glossary’s preamble. (Although in this case, there’s no logical order.) The result is shown in Example 138 on the next page.

Note that even though the duck and mallard entries have the `group` and `secondarygroup` fields set, there’s no group title for them in either glossary because they are child entries.

This means that only entries that don’t have a parent (with `flatten=false`) or any entry (with `flatten=true`) will have the group check performed. With `bib2gls`, the `group-level` will redefine `\glsxtraddgroup` to always do `⟨code⟩`, which means that all entries will have the group check performed.
8. Defining and Displaying Glossaries

Example 138: Displaying unsorted glossaries with custom groups

**Glossary**

<table>
<thead>
<tr>
<th>Group 1</th>
<th>Group 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>waterfowl</td>
<td>adder</td>
</tr>
<tr>
<td>ant</td>
<td>duck</td>
</tr>
<tr>
<td></td>
<td>gazelle</td>
</tr>
<tr>
<td></td>
<td>mallard</td>
</tr>
</tbody>
</table>

**Another Glossary**

<table>
<thead>
<tr>
<th>Group 1</th>
<th>Group 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>aardvark</td>
<td>ant</td>
</tr>
<tr>
<td></td>
<td>duck</td>
</tr>
<tr>
<td></td>
<td>zebra</td>
</tr>
</tbody>
</table>

If no group label has been provided no header will be added.

The above document can be modified to show the sub-group headings:

\renewcommand*{\glsxtraddgroup}[2]{#2}
\printunsrtglossary
\renewcommand*{\glsxtraddgroup}[2]{\ifnum\glscurrententrylevel<2 #2\fi}
\printunsrtglossary[type=another]

The result is shown in Example 139 on the following page.

Note that the mallard entry (which has hierarchical level 2) has its group shown in the first glossary (where the group is formed for all levels) but not in the second glossary (where the redefinition of \glsxtraddgroup restricts group formation to just level 0 and level 1).

There’s a small visual distinction between the group titles in different hierarchical levels
Example 139: Displaying unsorted glossaries with custom groups and sub-group headings

**Glossary**

- **Group 1**: zebra, aardvark
  - waterfowl
  - ant
- **Group 2**: gazelle
  - adder
- **Group 4**: duck

**Another Glossary**

- mallard
  - **Group 1**: aardvark
  - ant
  - zebra
- **Group 3**: duck
- **Group 2**: mallard

The top-level (level 0) groups have the title centred, whereas the sub-groups have their titles indented by the same amount as the corresponding sub-entries. This is due to the glossary style. Other styles may use the same formatting for all hierarchical levels.

The glossary styles provided with glossaries-extra and the base styles patched by glossaries-extra-stylemods all redefine:

\[\text{glssubgrouphheading\{\langle previous level\rangle\}\{\langle level\rangle\}\{\langle parent-label\rangle\}\{\langle group-label\rangle\}}\]

This command to format the sub-group headings in a manner applicable to the style. For example, styles that don’t show sub-entry names typically redefine this command to do nothing.

A default definition that simply does \[\text{glagrouphheading\{\langle group-label\rangle\}}\] is automatically initialised by \text{\setglossarystyle\{}\text{via \text\glsxtrpreglossarystyle\}}\} to allow for styles that don’t redefine this command. The first two arguments refer to the hierarchical level, where \text{\langle previous level\rangle} is the level of the previous group and \text{\langle level\rangle} is the level of this new sub-group.

---

393
8. Defining and Displaying Glossaries

The \textit{parent-label} is the label of the current entry’s parent, where the current entry is the first entry of the sub-group that immediately follows the heading.

The \texttt{bookindex} style defines \texttt{\gls{gls subgroup heading}} to use the style’s associated command \texttt{\glsxtrbookindexformatsubheader}. This can be redefined as required. For example, the following uses the parent entry’s hierarchical information:

\begin{verbatim}
\renewcommand*{\glsxtrbookindexformatsubheader}[5]{{
  \ifnum#2>1\relax
    \glstreesubsubitem\glstreegroupheaderfmt{\GlsXtrhiername{#3} / #5}\
  \else
    \glstreesubitem\glstreegroupheaderfmt{\GlsXtrhiername{#3} / #5}\
  \fi
}
\end{verbatim}

The above examples are contrived and demonstrate the need to define entries in a sensible order to achieve a sensible glossary with \texttt{\printunsrtglossary}. If you want to use this approach to display a glossary, you would need to make sure that you take care with the order that you define entries. This can be quite tedious for a large number of entries.

To switch to \texttt{bib2gls}, the entry data needs to be provided in a \texttt{bib} file. For example, the file \texttt{animalfamilies.bib} might contain:

\begin{verbatim}
@index{waterfowl, user1={Anseriformes}}
@index{ant, user1={Formicidae}}
@index{adder, user1={Vipera berus}}
@index{duck, parent={waterfowl}, user1={Anatidae}}
@index{zebra, user1={Hippotigris}}
@index{aardvark, user1={Orycteropus afer}}
@index{gazelle, user1={Gazella}}
@index{mallard, parent={duck}, user1={Anas platyrhynchos}}
\end{verbatim}

I’ve included some additional information stored in the \texttt{user1} field that wasn’t in the earlier examples. The document needs to use the \texttt{record} option and \texttt{\GlsXtrLoadResources} in order for it to work properly with \texttt{bib2gls}:

\begin{verbatim}
\usepackage[record, stylemods=bookindex, style=bookindex]{glossaries-extra}
\newglossary*{another}{Another Glossary}
\GlsXtrLoadResources{selection=all, \% select all entries
  src={animalfamilies}, \% identify bib file(s)
}\end{verbatim}
8. Defining and Displaying Glossaries

```latex
sort=en-GB,% sort method
secondary={la:user1:another}% sort again and copy to `another'
\glsdefpostname{index}{
  \emph{\glsentryuseri{\glscurrententrylabel}}}\}
\begin{document}
\printunsrtglossary
\printunsrtglossary[type=another]
\end{document}
```

If this code is saved in the file `myDoc.tex` then the build process is now:

```
pdflatex myDoc
bib2gls --group myDoc
pdflatex myDoc
```

The `--group` (or `-g`) switch is important as it instructs `bib2gls` to set the `group` field for the primary sort and the `secondarygroup` for the secondary sort. The primary sort will sort entries according to `en-GB` (British English). This can simply be set to `en` without a region. The secondary sort will resort the entries, but this time according to `la` (Latin) using the `user1` key as the sort value. The entry labels will then be copied to the custom `another` glossary.

The `glistex` file created by `bib2gls` (which will then be input by \GlsXtrLoadResources on the subsequent \LaTeX run) essentially contains the following code:

```
\glsxtrsetgrouptitle{6881280}{W}
\glsxtrsetgrouptitle{5832704}{G}
\glsxtrsetgrouptitle{5373952}{A}
\glsxtrsetgrouptitle{7077888}{Z}
\glsxtrsetgrouptitle{another5373952}{A}
\glsxtrsetgrouptitle{another5767168}{F}
\glsxtrsetgrouptitle{another6356992}{O}
\glsxtrsetgrouptitle{another5898240}{H}
\glsxtrsetgrouptitle{another6815744}{V}
\glsxtrsetgrouptitle{another5832704}{G}

\longnewglossaryentry*[aardvark]{name={aardvark},
  user1={Orycteropus afer},group={5373952}}{}
\longnewglossaryentry*[adder]{name={adder},
  user1={Vipera berus},group={5373952}}{}
\longnewglossaryentry*[ant]{name={ant},
```
It's more complicated than this as helper commands are provided to make it easier to customize and the entries will all have category={index} since they were defined with @index, but this is basically like the preamble in the earlier examples, except that the ordering and groups are more logical. The result is shown in Example 140 on the next page.

Note that the group and secondarygroup fields haven't been set for the child entries (duck and mallard). This is the default behaviour and it means that regardless of the definition you provide for \glsxtraddgroup, sub-groups won't be displayed. If you want those fields set for child entries, you need to use the group-level resource option. For example:

\GlsXtrLoadResources[selection=all,\% select all entries
group-level={<1},\% level 0 and 1
src={animalfamilies},\% identify bib file(s)
8. Defining and Displaying Glossaries

Example 140: Displaying sorted glossaries with groups using bib2gls

Glossary

A
aardvark (Orycteropus afer)
adder (Vipera berus)
ant (Formicidae)

G
gazelle (Gazella)

W
waterfowl (Anseriformes)
duck (Anatidae)
mallard (Anas platyrhynchos)

Z
zebra (Hippotigris)

Another Glossary

A
waterfowl (Anseriformes)
duck (Anatidae)
mallard (Anas platyrhynchos)

F
ant (Formicidae)

G
gazelle (Gazella)

H
zebra (Hippotigris)

O
aardvark (Orycteropus afer)

V
adder (Vipera berus)

This will add support for level 0 (no parent) and level 1 (parent but no grandparent) entries. Deeper levels won’t have support. The --group switch is still required.

8.4.2. Location Lists

The “unsrt” family of commands check for the existence of the location and loclist keys. These are both defined by the record option. (The loclist field is also used by \makeindex but isn’t defined as a key.)
8. Defining and Displaying Glossaries

The `location` field (if set) should contain the formatted location list. This is checked first and used if not empty. Otherwise the `loclist` field (if set) is used, but that will use the same method as \printnoidxglossary to format, which doesn’t compact consecutive locations.

It’s possible to choose a different field for the formatted location list by redefining:

```
\GlsXtrLocationField initial: location
```

This should expand to the internal field label. If the field is not the default `location` then the test for `loclist` is omitted.

Whichever field is used, the formatted location list is passed to the appropriate glossary style command (\glossentry or \subglossentry) encapsulated with \glossaryentrynumbers.

If there’s no location field or if the tested fields are empty, then an empty argument (with no \glossaryentrynumbers encapsulator) is passed to the glossary style command. In this case, the nonumberlist option is redundant as there’s no location list to suppress.

8.4.3. Advanced Commands

To provide a better understanding of how filtered and inner glossaries work, it’s useful to understand the difference between \printglossary (used with makeindex and xindy) and \printunsrtglossary (used with bib2gls).

In the first case, makeindex or xindy is used to create a file that contains content in the form:

```
glossarysection{\glossarytocitle}{\glossarytitle}%
glossarypreamble
\begin{theindex}
glossaryheader
(content)
\end{theindex}
glossarypostamble
```

where `content` contains lines such as:

```
glsgroupheading{⟨group label⟩}\relax \glsresetentrylist
\glossentry{⟨entry label⟩}{⟨location list⟩}%
\subglossentry{⟨level⟩}{⟨entry label⟩}{⟨location list⟩}%
```

The group headings (see §8.4.1) are typeset using \glsgroupheading. Top-level entries are typeset with \glossentry and child entries are typeset with \subglossentry where `⟨level⟩` indicates the hierarchical level. Both makeindex and xindy order the items so that the child entries are placed immediately after the corresponding parent entry.

The \printglossary command essentially does:
8. Defining and Displaying Glossaries

\texttt{(Set default title and style.)}
\texttt{\textbackslash bgroup}
\texttt{(Initial setup.)}
\texttt{(Input the file created by makeindex or xindy.)}
\texttt{\textbackslash egroup}

The initial setup part sets the glossary style (which determines the definitions of the glossary, \texttt{\glossaryheader}, \texttt{\glsgroupheading}, \texttt{\glossentry} and \texttt{\subglossentry}), assigns the title \texttt{(\glossarytitle and \glossarytoctitle)} and defines \texttt{\currentglossary}. (There is some other stuff done both before and after the file is input, but that’s not relevant here.)

In the case of \texttt{bib2gls}, there isn’t a glossary file to input. Instead, \texttt{bib2gls} is used to create a file that contains the entry definitions, which is input in the document preamble (via \texttt{\GlsXtrLoadResources}). The entries are defined in the required order and use internal fields to store the indexing information (such as the group label and location lists). Now \texttt{\printunsrtglossary} is used to display the glossary, which essentially does:

\texttt{(Set default title and style.)}
\texttt{\textbackslash bgroup}
\texttt{(Initial setup.)}
\texttt{\glossarysection[\glossarytoctitle]{\glossarytitle}¥}
\texttt{\glossarypreamble}
\texttt{(Construct internal control sequence containing glossary content.)}
\texttt{(Expand internal control sequence.)}
\texttt{\glossarypostamble}
\texttt{\textbackslash egroup}

The \texttt{(initial setup)} is the same as for \texttt{\printglossary}. The key difference here is that there’s no file containing the typeset glossary that can be simply input. Instead it’s necessary to iterate over the glossary’s internal label list. Some of the glossary styles use a tabular-like environment (such as \texttt{longtable}, which is used by the \texttt{long} styles). It’s always problematic having a loop inside a tabular context so \texttt{\printunsrtglossary} by-passes the problem by moving the loop outside of the \texttt{theglossary} environment. The command iterates over all entry labels (in the order in which they were added to the glossary) and constructs an internal control sequence (\texttt{\@glsxtr@doglossary}), which ends up containing:

\texttt{\begin{theglossary}\glossaryheader\glsresetentrylist
(content)}
\texttt{\end{theglossary}}
The \textit{content} in this case is different as it doesn’t explicitly contain \texttt{\glossentry} and \texttt{\subglossentry} but instead uses an internal handler that just takes the entry label as the argument. The \texttt{\glsgroupheading{⟨group label⟩}} command is inserted whenever a top-level entry has the group field set to a label that’s different to the previous top-level entry’s group field (and, if supported, sub-groups are similarly inserted with \texttt{\glssubgroupheading}, see §8.4.1). So the content is in the form:

\begin{verbatim}
\glsgroupheading{⟨group label⟩}
\langle internal cs handler\rangle{⟨entry label⟩}
\langle internal cs handler\rangle{⟨entry label⟩}
...
\glsgroupheading{⟨group label⟩}
\langle internal cs handler\rangle{⟨entry label⟩}
\langle internal cs handler\rangle{⟨entry label⟩}
...
\end{verbatim}

There are hooks and commands available for use within those hooks that may be adjusted to customize the way the glossary is displayed. These are described below.

At each iteration (while the glossary content is being constructed), the following steps are performed:

1. Store the current entry label in \texttt{\glscurrententrylabel}.
2. If \texttt{\glscurrententrylabel} is empty, skip this iteration.
3. Define placeholder commands:

\begin{verbatim}
\glscurrententrylevel
\glscurrenttoplevelentry
\glscurrentrootentry
\end{verbatim}

This is set to the current entry’s hierarchical level (taking \texttt{leveloffset} and \texttt{flatten} options into account);

This is set to the current entry label if \texttt{\glscurrententrylevel} is 0 (that is, it expands to the most recent top-level entry, allowing for \texttt{flatten} and \texttt{leveloffset});
8. Defining and Displaying Glossaries

This is set to the current entry label if flatten=true or if the current entry doesn’t have a parent (that is, it expands to the most recent top-level entry, allowing for flatten but not leveloffset).

4. Perform the entry process hook:

\printunsrtglossaryentryprocesshook\{⟨entry-label⟩\}

This does nothing by default. Within the definition of this hook, you may use:

\printunsrtglossaryskipentry

This will cause the remainder of the current iteration to be skipped, which will prevent the current entry from being shown in the glossary.

5. If groups=true, use \glsxtraddgroup (see §8.4.1) to append the top-level group heading (\glsgroupheading) or the sub-group heading (\glssubgroupheading) to \@glsxtr@doglossary.

6. Append \langle internal cs handler\rangle\{⟨entry label⟩\} to \@glsxtr@doglossary.

The placeholders \glscurrenttoplevelentry and \glscurrentrootentry may not be an ancestor of the current entry. For example, if the glossary doesn’t have child entries immediately following their parent entry.

Once the glossary construction (\@glsxtr@doglossary) has been completed, the following hook is performed:

\printunsrtglossarypredoglossary

This does nothing by default. You can redefine this to show the definition of \@glsxtr@doglossary for debugging purposes:

\renewcommand{\printunsrtglossarypredoglossary}{\%\csshow{\@glsxtr@doglossary}}

This will interrupt the \LaTeX run and display the definition in the transcript.

The handler command \langle internal cs handler\rangle performs the following:
8. Defining and Displaying Glossaries

\protected@xdef\glscurrententrylabel{⟨entry-label⟩}\
\printunsrtglossaryhandler\glscurrententrylabel

This stores the entry’s label in \glscurrententrylabel (which allows it to be referenced in style hooks, such as the post-name hook or post-description hook). Note that it uses a global definition to avoid scoping issues caused with tabular-like styles. The main handling of the entry is performed by:

\printunsrtglossaryhandler{⟨entry-label⟩}

This is simply defined to use:

\glsxtrunsrtdo{⟨entry-label⟩}

This displays the entry according to the current glossary style, taking the hierarchical level into account (as given by \glscurrententrylevel).

The following are additional commands that may be useful in the above hooks.

\glsxtriflabelinlist{⟨label⟩}{⟨label-list⟩}{⟨true⟩}{⟨false⟩}

Does ⟨true⟩ if the given label is in the given comma-separated list of labels, otherwise does ⟨false⟩. The label and list are fully expanded.

\ifglsxtrprintglossflatten ⟨true⟩\else ⟨false⟩\fi initial: \iffalse

This conditional is set by the flatten option and can be used to test if the option has been set.

For example, the following skips all entries that have the category set to symbol:

\usepackage[style=index]{glossaries-extra}
\newglossaryentry{ant}{name={ant},description={}}
\newglossaryentry{pi}{name={}\ensuremath{\pi},description={},category={symbol}}
\newglossaryentry{aardvark}{name={aardvark},description={}}
\newglossaryentry{alpha}{name={}\ensuremath{\alpha},description={},category={symbol}}
\begin{document}
8. Defining and Displaying Glossaries

Example 141: Filtering by category

Glossary

ant
aardvark

8.4.3.1. Inner Glossaries

See also Gallery: Inner or Nested Glossaries.\(^*\)

\(^*\)dickimaw-books.com/gallery/index.php?label=bib2gls-inner

It’s possible you may want to combine multiple glossaries sequentially, as sub-blocks of a single list. The inner part of \printunsrtglossary can be created with:

\printunsrtinnerglossary[⟨options⟩]{⟨pre-code⟩}{⟨post-code⟩}

This can’t be used on its own, as it only forms a fragment. It doesn’t include the section header, style initialisation, preamble, \theglossary environment, header and postamble.

As with \printunsrtglossary, \printunsrtinnerglossary constructs an internal control sequence containing the content, but it adds scoping to localise the effects of any supplied options. So it essentially does:

\begingroup

(Initial setup (process options).)
⟨pre-code⟩
(Construct internal control sequence containing glossary content.)
⟨Expand internal control sequence.⟩
⟨post-code⟩

\endgroup
There are two ways this command may be used.

\begin{printunsrtglossarywrap}[(options)]
\langle\textit{content}\rangle
\end{printunsrtglossarywrap}

The \texttt{printunsrtglossarywrap} environment takes one optional argument that uses the same keys as \texttt{printunsrtglossary} (see §8.3). Note that in this case the \texttt{type} key simply provides a title (if one has been assigned to that glossary). It doesn’t indicate the content. There’s no point using both \texttt{type} and \texttt{title}.

The start of the environment sets up the glossary style and does the header:

\begin{quote}
\textit{Set default title and style.}
\textit{Initial setup.}
\end{quote}
\glossarysection[\glossarytoctitle]{\glossarytitle}%
\glossarypreamble
\begin{theglossary}
\glossaryheader\glsresetentrylist
\end{theglossary}
\glossarypostamble

The end of this wrapper environment ends the \texttt{glossary} and does the postamble:

\end{theglossary}\glossarypostamble

For example, the following has two glossaries but displays them as inner glossaries:

\begin{document}
\begin{printunsrtglossarywrap}[style=index]
\glstreeitem First Glossary
\printunsrtinnerglossary[leveloffset=1]{\}{\}{\}
\glstreeitem Second Glossary
\printunsrtinnerglossary[type=other, leveloffset=1]{\}{\}{\}
\end{printunsrtglossarywrap}
\end{document}
8. Defining and Displaying Glossaries

Example 142: Inner glossaries using \glossarywrap

\begin{flushleft}
\textbf{Glossary}
\begin{description}
\item[First Glossary]
\begin{itemize}
\item ant
\item bee
\end{itemize}
\item[Second Glossary]
\begin{itemize}
\item duck
\item goose
\end{itemize}
\end{description}
\end{flushleft}

The other way that \glossaryinner can be used is within \glossary. The handler function described in §8.4.3 that’s used to process each entry to be displayed in the glossary, is defined as:

\newcommand{\printglossaryhandler}[1]{\glsxtrndo{#1}}

It’s possible to redefine this so that it also displays an inner glossary. The following example has the terms “pictograph” and “Greek symbol” in the main glossary. Two ignored glossaries are created (which don’t require a title) where the glossary label matches an entry label in the main glossary.

\begin{verbatim}
\usepackage{fontawesome}
\usepackage[style=tree]{glossaries-extra}
\newglossaryentry{pictograph}{name={pictograph},
   description={picture or symbol representing a word or phrase}}
\newglossaryentry{mathgreek}{name={Greek symbol},
   description={mathematical constants or functions}}
\newignoredglossary{pictograph}
\newglossaryentry{cut}{type={pictograph},
   name={\faCut},
   description={cut}}
\newglossaryentry{paste}{type={pictograph},
   name={\faPaste},
   description={paste}}
\newglossaryentry{alpha}{type={mathgreek},
   name={\ensuremath{\alpha}},
\end{verbatim}
8. Defining and Displaying Glossaries

This creates a custom command \texttt{\nestedhandler} that can be used as the handler to create nested glossaries. After each item in the glossary, if the entry’s label matches the label of a defined glossary, that glossary is displayed with its hierarchical level incremented by 1, which creates the illusion of child entries. The resulting document is shown in Example 143.

Example 143: Nested glossaries

**Glossary**

<table>
<thead>
<tr>
<th>pictograph</th>
<th>picture or symbol representing a word or phrase</th>
</tr>
</thead>
<tbody>
<tr>
<td>☢️</td>
<td>cut</td>
</tr>
<tr>
<td>☑️</td>
<td>paste</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Greek symbol</th>
<th>mathematical constants or functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>α</td>
<td>alpha</td>
</tr>
<tr>
<td>β</td>
<td>beta</td>
</tr>
</tbody>
</table>

8.4.3.2. Per-Unit Glossaries

If you are using bib2gls then it’s possible to only list entries that match a particular counter value. For example, you may want a mini-glossary at the start of a section that only lists the entries that have been recorded in that section. This can be done by using the handler to skip entries that don’t have a matching record. It can also be implemented with record counting, as shown in Example 156 on page 533 in §11.4.

It’s also possible to make each indexing instance automatically make a note of a particular
8. Defining and Displaying Glossaries

counter using:

\GlsXtrRecordCounter{⟨counter-name⟩}

(This doesn’t correspond to a \bib2gls record. That’s dealt with by the indexing that comes first.)

This command may only be used in the preamble (with record) and indicates that whenever an entry is indexed, the following line should be added to the aux file:

\glsxtr@counterrecord{⟨entry-label⟩}{⟨counter⟩}{⟨value⟩}

where ⟨value⟩ is given by \the⟨counter⟩. On the next \LaTeX run, this information is picked up from the aux file and the information is added to the record.⟨counter⟩ field (stored as an etoolbox internal list). This internal command is only used in the aux file and has a user-level hook:

\glsxtrAddCounterRecordHook{⟨entry-label⟩}{⟨counter⟩}{⟨value⟩}

This does nothing by default. If you want to redefine this, the redefinition must be placed in the document preamble before the aux file is input.

There are two ways of skipping an entry. The first is to redefine \printunsrtglossary-entryprocesshook to perform the test and use \printunsrtglossaryskipentry to skip an unwanted entry (as illustrated earlier). The second is to perform the test in \printunsrtglossaryhandler. The first method is the better option for large lists that may contain group headers. The example below uses the second method.

The file myentries.bib contains the following:

@symbol{pi,name={\ensuremath{\pi}},
description={ratio of the length of the circumference of a circle
to its diameter}}
@symbol{root2,name={\ensuremath{\sqrt{2}}},
description={Pythagoras' constant}}
@symbol{zeta3,name={\ensuremath{\zeta(3)}},
description={Ap\'ery's constant}}
@symbol{zero,name={0},
description={nothing or nil}}
@symbol{one,name={1},
description={single entity, unity}}

The document redefines the handler to only show entries in the current section:
\usepackage[record,stylemods,style=index]{glossaries-extra}
\GlsXtrRecordCounter{section}
\GlsXtrLoadResources[src={myentries}]
\begin{document}
\renewcommand{\printunsrtglossaryhandler}{1}{\%
  \glsxtrfieldxifinlist{#1}{record.section}{\thesection}
  {\glsxtrunsrtdo{#1}}{}
}%
\section{Sample}
\printunsrtglossary
This section discusses \gls{pi}, \gls{root2} and \gls{zeta3}.

\section{Another Sample}
\printunsrtglossary
This section discusses \gls{one}, \gls{pi} and \gls{zero}.
\end{document}

If the document is saved in the file myDoc.tex then the build process is:

```
pdflatex myDoc
bib2gls myDoc
pdflatex myDoc
```

The first \LaTeX run adds the records to the aux file for bib2gls to pick up, but also adds the \glsxtr@counterrecord lines (which bib2gls ignores) that setup the \texttt{record.section} list field for the given entry.

This means that \texttt{\glsxtrfieldxifinlist} can be used to determine whether or not the current section number (\texttt{\thesection}) is in the list. If it is, then the entry is displayed in the current glossary style using the default \texttt{\glsxtrunsrtdo}. Otherwise nothing is displayed.

The following command is provided that performs something similar:

\texttt{\printunsrtglossaryunit[(options)]{(counter-name)}}

This is equivalent to:


This initialises the hook via:

\printunsrtglossaryunitsetup{(counter-name)}

This is essentially does:

\% redefine handler to only show entries with a match:
\renewcommand{\printunsrtglossaryhandler}[1]{%
  \glsxtrfieldxifinlist{#1}{record.(counter-name)}{\the(counter-name)}
  {\glsxtrunsrtdo{#1}}%
}%
\% assign target name prefixes (see below)
\% suppress section header:
\renewcommand*{\glossarysection}[2]{%}
\% append vertical space after the glossary:
\appto\glossarypostamble{\printunsrtglossaryunitpostskip}%

This is more complicated than the original example as it also suppresses the glossary section header and modifies the target name prefix. Additionally, the following is appended to the end of the glossary:

\printunsrtglossaryunitpostskip

This simply does:

\glspar\medskip\glspar

which creates a small vertical space. The target name prefix (targetnameprefix) is assigned as follows. If \the(counter-name) has been defined, the prefix is:

record.(counter-name).\the(counter-name).\@gobble

otherwise the prefix is:

record.(counter-name).\the(counter-name).\@gobble

The use of \@gobble at the end discards \glolinkprefix.
The above example can be rewritten using \printunsrtglossaryunit. I’ve added symbol-sort-fallback to sort by the description and a full glossary at the end of the document.

\usepackage[record,stylemods,style=index]{glossaries-extra}
\GlsXtrRecordCounter{section}
\GlsXtrLoadResources[src={myentries},
symbol-sort-fallback=description]
\begin{document}
\section{Sample}
\printunsrtglossaryunit{section}
This section discusses \gls{pi}, \gls{root2} and \gls{zeta3}.
\section{Another Sample}
\printunsrtglossaryunit{section}
This section discusses \gls{one}, \gls{pi} and \gls{zero}.
\printunsrtglossaries
\end{document}

The build process is the same as before:

pdflatex myDoc
bib2gls myDoc
pdflatex myDoc

The resulting document is shown in Example 144 on the following page. Note that all glossaries show the location lists, which all contain the page number 1, since the example document is only one page long.

Other variations include creating a secondary glossary that’s ordered differently for the mini-glossaries. For example:

\newignoredglossary*{glossary2}
\GlsXtrLoadResources[src={\jobname},
symbol-sort-fallback=description,
secondary=use:glossary2]

This orders the secondary glossary according to use (the first record for the entire document...
8. Defining and Displaying Glossaries

Example 144: Sub-glossary for a given counter value

1 Sample

ζ(3) Apéry’s constant 1
√2 Pythagoras’ constant 1
π ratio of the length of the circumference of a circle to its diameter 1

This section discusses π, √2 and ζ(3).

2 Another Sample

0 nothing or nil 1
π ratio of the length of the circumference of a circle to its diameter 1
1 single entity, unity 1

This section discusses 1, π and 0.

Glossary

ζ(3) Apéry’s constant 1
0 nothing or nil 1
√2 Pythagoras’ constant 1
π ratio of the length of the circumference of a circle to its diameter 1
1 single entity, unity 1

not for the given unit). The mini-glossaries will then need the type option:

\printunsrtglossaryunit[type=glossary2]{section}

There is an alternative method that ensures the mini-glossaries are ordered by use within the section. This can be done by redefining \glsxtrAddCounterRecordHook to create a glossary for each unit (instead of using a secondary glossary):

\renewcommand{\glsxtrAddCounterRecordHook}[3]{%
  \provideignoreglossary{#2.#3}%
  \glsxtrcopytoglossary*{#1}{#2.#3}%
}

411
8. Defining and Displaying Glossaries

(Remember this needs to be done in the preamble, before the aux file is input.)

This creates a glossary with the label \langle counter \rangle \langle value \rangle, if it’s not already defined, and adds the entry’s label to it. This means that this glossary will only contain the entries for the matching \langle counter \rangle and \langle value \rangle, and the entry labels are in the order they were added to the aux file.

The glossary needs to be set appropriately. For example:

\printunsrtglossaryunit[type=section.\thesection]{section}

There’s now no filtering required, but \printunsrtglossaryunit is still useful as it automatically suppresses the section header, alters the hyperlink prefix and adds extra spacing after the glossary. However, if you prefer, you can simply do something like:

\printunsrtglossary*[type=section.\thesection, target=false]
\renewcommand*{\glossarysection}[2][{}]{}
\printunsrtglossaryunitpostskip

This is done in Example 145 on the next page.

8.5. Standalone Entry Items

It may be that you don’t want a list but would rather display entry details throughout the document. You can simply do \glsentryname followed by \glsentrydesc. (Remember that if you don’t want a sorted list, use sort=none or sort=clear to skip the preprocessing of the sort field.) For example, in the preamble provide a custom command to display the entry’s name and description:

\newcommand{\displayterm}[1]{%
\par\medskip\par\noindent
Definition: \glsentryname{#1}.\par
\glsentrydesc{#1}
\par\medskip
}

define your entries, for example:
Example 145: Sub-glossary for a given counter value ordered by use in the section

1 Sample

π ratio of the length of the circumference of a circle to its diameter 1
\sqrt{2} Pythagoras’ constant 1
ζ(3) Apéry’s constant 1

This section discusses π, \sqrt{2} and ζ(3).

2 Another Sample

1 single entity, unity 1
π ratio of the length of the circumference of a circle to its diameter 1
0 nothing or nil 1

This section discusses 1, π and 0.

Glossary

ζ(3) Apéry’s constant 1
0 nothing or nil 1
\sqrt{2} Pythagoras’ constant 1
π ratio of the length of the circumference of a circle to its diameter 1
1 single entity, unity 1

\newglossaryentry{function}{name={function},
\quad\text{description}={a \ \text{relation} \ \text{or expression involving variables}}}

and then later in the text:

\displayterm{function}

However, if may be that you want to use hyperref and have commands like \citet link back
8. Defining and Displaying Glossaries

to the place where the term is described. Instead of using \glsentryname use:

\glsxtrglossentry{(entry-label)}

where (entry-label) is the entry’s label.

This is designed to behave much like the way the name is displayed in the glossary. It performs the following:

- Defines \glscurrententrylabel to the entry’s label. This is usually done at the start of the glossary style commands \glossentry and \subglossentry and may be used by hooks, such as the post-name hook. Here the definition is localised so that it’s only available for use in \glossentryname.

- Defines \currentglossary to the entry’s glossary type. This is usually done at the start of commands like \printglossary and may be used by style hooks. Here the definition is localised so that it’s only available for use in \glsentryitem and \gls-subentryitem. The value is obtained by fully expanding:

\GlsXtrStandaloneGlossaryType

which defaults to the value of the type field for the current entry.

- Increments and display the entry counters if the entrycounter or subentrycounter package options are set. If the entry doesn’t have a parent, then this does:

\glsentryitem{(entry-label)}

otherwise it does:

\GlsXtrStandaloneSubEntryItem{(entry-label)}

which defaults to \glssubentryitem{(entry-label)} if the entry has a parent but not a grandparent.

This reflects the behaviour of the predefined hierarchical styles. A bug in pre-version 1.31 used \glsentryitem for all child levels, which doesn’t match the hierarchical glossary styles. If you want to restore this behaviour, just do:

\renewcommand*\GlsXtrStandaloneSubEntryItem[1]{% \glssubentryitem[#1]}
• Sets the hyper-target if supported (using \glstarget) and displays the entry name using:

\GlsXtrStandaloneEntryName{⟨entry-label⟩}

which uses \glstarget{⟨entry-label⟩}{\glossentryname{⟨entry-label⟩}} by default. Remember that \glossentryname uses \glsnamefont or picks up the style from category attributes such as glossnamefont.

If you have used \nopostdesc or \glsxtrnopostpunc in any of your description fields, you can use:

\glsxtractivatenopost

to make these commands behave as they normally do within a glossary. This needs to be placed before:

\glossentrydesc{⟨entry-label⟩}\glspostdescription

and scoped. Note that \glsnonextpages and \glsnextpages have no effect outside of the glossary and are not intended for use in a standalone context.

It’s also possible to select a different field (rather than using name):

\glsxtrglossentryother{⟨header⟩}{⟨entry-label⟩}{⟨field-label⟩}

The ⟨field-label⟩ must be given using its internal field label. The ⟨header⟩ argument is the code to pass to the third argument of \glsxtrtitleorpforheading. It may be left empty in which case the default is determined as follows:

• If \glsxtrhead{field-label} is defined (see §5.3.3), then ⟨header⟩ is \glsxtrhead{field-label}{entry-label}.

• Otherwise ⟨header⟩ is simply the field value.

The \glsxtrglossentryother command internally uses

\GlsXtrStandaloneEntryOther{⟨entry-label⟩}{⟨field-label⟩}

instead of \GlsXtrStandaloneEntryName, which uses \glossentrynameother{⟨entry-label⟩}{field-label} instead of \glossentryname{⟨entry-label⟩}.
8. Defining and Displaying Glossaries

If you have loaded the glossaries-accsupp package (through the accsupp option) then accessibility support will be provided if there’s a corresponding command:

\gls\langle field-label\rangle\text{accessdisplay}\{(text)\}\{(entry-label)\}

(for example, \glssymbolaccessdisplay).

This means that my custom command can be changed to:

\newcommand{\displayterm}[1]{\%  
\par\medskip\par\noindent  
Definition: \glsxtrglossentry{#1}.\par  
\glsentrydesc{#1}  
\par\medskip\par\medskip\par\medskip\par\medskip\par
}

If I want numbered definitions, then I can use the package options entrycounter or subentrycounter and remove the colon:

\newcommand{\displayterm}[1]{\%  
\par\medskip\par\noindent  
Definition \glsxtrglossentry{#1}.\par  
\glsentrydesc{#1}  
\par\medskip\par\medskip\par\medskip\par\medskip\par
}

The counter label uses a dot after the number by default but this can be changed to a colon:

\renewcommand*{\glsentrycounterlabel}{\theglossaryentry:\space}

It’s now possible to not only use \gls to link back to the definition but also use \glsrefentry to reference the counter and \glsxtrpageref to reference the page number.

If I want the description to behave more like it does in a glossary in need to make the following modification:

\newcommand{\displayterm}[1]{\%  
\par\medskip\par\noindent  
Definition \glsxtrglossentry{#1}.\par  
\begingroup \glsxtractivatenopost

416
8. Defining and Displaying Glossaries

(Note the grouping to localise \glsxtractivatenopost.)

You can also use \glsxtrglossentry within section headings. For example:

\section{\glsxtrglossentry{function}}

If \glsxtrglossentry occurs in a section title and hyperref has been loaded, then \glsxtrglossentry will expand in the PDF bookmark as:

\GlsXtrStandaloneEntryPdfName{⟨entry-label⟩}

This defaults to \glsentryname{⟨entry-label⟩}. The page headers and table of contents will use

\GlsXtrStandaloneEntryHeadName{⟨entry-label⟩}

which defaults to \glsxtrheadname{⟨entry-label⟩}. For example, to ensure that the name is displayed in sentence case in the title, PDF bookmarks and heading:

\glssetcategoryattribute{general}{glossname}{firstuc}
\renewcommand{\GlsXtrStandaloneEntryPdfName}{[1]{\Glsentryname{#1}}}
\renewcommand{\GlsXtrStandaloneEntryHeadName}{[1]{\Glsentryname{#1}}}

Note that this requires glossaries v4.50+ to ensure that \Glsentryname expands. An alternative is to use \Glsxtrusefield.

If \glsxtrglossentryother occurs in a section title and hyperref has been loaded, then \glsxtrglossentryother will expand in the PDF bookmark as:

\GlsXtrStandaloneEntryPdfOther{⟨entry-label⟩}{⟨field-label⟩}

This defaults to the value of the given field. The page headers and table of contents will use
8. Defining and Displaying Glossaries

the \textit{header} argument, if not empty, otherwise it will use:

\GlsXtrStandaloneEntryHeadOther{⟨entry-label⟩}{⟨field-label⟩}

This does \texttt{\glsxtrhead}⟨field-label⟩, if it exists, or otherwise it just does the value of the given field (which can be obtained with \texttt{\glsxtrusefield}).

If you’re using a page style or table of contents that doesn’t use \texttt{\markright} or \texttt{\markboth} or \texttt{@starttoc} then you need to insert \texttt{\glsxtrmarkhook} and \texttt{@glsxtrinmark} at the start of the header or table of contents either scoped or afterwards cancelled with \texttt{@glsxtrnotinmark} and \texttt{\glsxtrrestoremarkhook}, see §5.3.3.

8.6. Glossary Style Modifications

The \texttt{glossaries-extra} package redefines \texttt{\setglossarystyle}, and it now includes a hook that’s performed before the style is set:

\glsxtrpreglossarystyle

This allows for new style commands that aren’t provided by the base \texttt{glossaries} package to be initialised in the event that a style that doesn’t redefine them is used. The default definition is:

\newcommand{\glsxtrpreglossarystyle}{%
\renewcommand*{\glssubgroupheading}[4]{\glsgroupheading{##4}}%
}

If you prefer a different default, you can redefine this command as appropriate.

The commands \texttt{\glossentryname} and \texttt{\glossentrydesc} are modified to take into account the \texttt{glossname}, \texttt{glossnamefont}, \texttt{glossdesc} and \texttt{glossdescfont} attributes (see §10). This means you can make simple font or case-changing modifications to the name and description without defining a new glossary style.

The command \texttt{\glossentriesymbol} is modified to take into account the \texttt{glosssymbolfont} attribute. Note that, unlike the above, there’s no corresponding attribute to change the case as it’s usually not appropriate to change the case of a symbol (and for some symbols, such as pictographs, there’s no concept of case). If \texttt{\texorpdfstring} has been defined \texttt{\glossentriesymbol} will be defined to do:

\texttt{\texorpdfstring}{⟨\LaTeX\ code⟩}{⟨PDF⟩}
The ⟨\LaTeX\ code⟩ part is robust and deals with the actual typesetting of the symbol. The ⟨PDF⟩ part is simply:

\glsentrypdfsymbol{⟨entry-label⟩}

which is defined to just do \glsentriesymbol{⟨entry-label⟩}. The chances are that the code in the symbol key won’t be valid in the PDF bookmarks, so you can redefine \glsentrypdfsymbol to use a more appropriate field. (If you do redefine this command, remember that it needs to fully expand.)

For example, if you are using glossaries-accsupp, you could use the symbolaccess field:

\renewcommand{\glsentrypdfsymbol}{% \glsentrysymbolaccess{#1}}

Alternatively, if you are using bib2gls you can use the \LaTeX parser library to interpret a copy of the symbol field and use that. For example, with the resource options:

\renewcommand{\glsentrypdfsymbol}{% \glsentryuseri{#1}}

(You may need \LaTeX\ or Lua\LaTeX\ with this method.) This allows \glossentriesymbol to be used in a section heading with standalone definitions. See the bib2gls manual for further details about the \LaTeX interpreter.

If you want to adapt a style to use another field instead of name, you can use:

\glossentrynameother{⟨entry-label⟩}{⟨field-label⟩}

This behaves just like \glossentryname (that is, it obeys the glossname attribute, uses either the glossnamefont attribute or \glsnamefont to format the text, and uses the post-name hook) but the text is obtained from the field given ⟨field-label⟩ instead of name. The ⟨field-label⟩ argument must be the internal field label (for example desc rather than description).
8. Defining and Displaying Glossaries

8.6.1. Post-Name Hooks

The glossaries-extra package adds a hook to \glossentryname and \Glossentryname (which is used in glossary styles to display the entry’s name):

\glsxtrpostnamehook{(entry-label)}

This is the main post-name hook, which implements additional hooks to allow for customisation. By default, \glsxtrpostnamehook checks the indexname attribute. If the attribute exists for the category to which the entry belongs, then the name is automatically indexed using:

\glsxtrdoautoindexname{(entry-label)}{indexname}

See §12 for further details.

The post-name hook \glsxtrpostnamehook will also use:

\glsxtrpostname\langle category \rangle

if it exists. You can use \glscurrententrylabel to obtain the entry label with the definition of this command. For example, suppose you are using a glossary style that doesn’t display the symbol, you can insert the symbol after the name for a particular category, say, the “symbol” category:

\newcommand*{\glsxtrpostnamesymbol}{\space (\glsentrysymbol{\glscurrententrylabel})}

For convenience, you can use:

\glsdefpostname{(category)}{(definition)}

This is simply a shortcut for:

\csdef{\glsxtrpostname\langle category \rangle}{\langle definition \rangle}

Note that it doesn’t check if the command has already been defined.
The post-name hook also does:

\glsextrapostnamehook\{\textit{entry-label}\}

(before \glsxtrpostname\{\textit{category}\}) to allow for additional non-category related code. This does nothing by default.

### 8.6.2. Post-Description Hooks

The glossaries package provides the hook \glspostdescription, which is placed after the description in some of the predefined styles. The glossaries-extra-stylemods package modifies the predefined styles to ensure that they all use this hook. This provides a convenient way to make slight adjustments, such as appending content after the description, without having to define a custom glossary style.

The glossaries-extra package redefines \glspostdescription so that it includes the following hook:

\glsxtrpostdescription

This new hook simply performs the category post-description hook:

\newcommand*{\glsxtrpostdescription}{%
\csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%}

The punctuation that is automatically inserted with \texttt{postdot} or \texttt{postpunc} is placed after \glsxtrpostdescription, not before.

If you want to modify the hook for all entries (without affecting the \texttt{postpunc} or \texttt{postdot} options), then redefine \glsxtrpostdescription. If you want to adjust this hook according to the entry’s category, then you can simply redefine the category post-description hook.

\glsxtrpostdesc\{\textit{category}\}

Some common category post-description hooks are provided:
The post-description hook for the `general` category.

The post-description hook for the `term` category.

The post-description hook for the `acronym` category.

The post-description hook for the `abbreviation` category.

The above all do nothing by default. You can redefine them with \renewcommand or use:

\glsdefpostdesc{⟨category⟩}{⟨definition⟩}

This will define (or redefine) \glsxtrpostdesc{category}. The package options `symbols`, `numbers` and `index` provide corresponding category post-description hooks.

You can reference the current entry within these hooks using \glscurrententrylabel, which is defined within the glossary (any of the \print…glossary commands) and also within the standalone commands, such as \glsxtrglossentry.

Suppresses the post-description punctuation that is automatically inserted by package options `postdot` or `postpunc`.

The glossaries package provides \nopostdesc, which may be used in the `description` to suppress the post-description hook for that entry. This suppresses both the post-description punctuation and the additional \glsxtrpostdescription hook. If you only want to suppress to punctuation, then use \glsxtrnopostpunc instead.

The post-description hooks are implemented by \glspostdescription within the glossary style. If this command isn’t used in the style, then the additional hooks won’t
8. Defining and Displaying Glossaries

If this command is placed in the definition of \glsxtrpostdescription or added to the category post-link hook, then it will counter-act any use of \glsxtrnopostpunc to restore the post-description punctuation.

These commands have no effect outside of the glossary (except with standalone entries that use \glsxtractivatenopost and \glspostdescription, see §8.5).

8.6.3. Number (Location) List

The location list is now placed inside the argument of:

\GlsXtrFormatLocationList{⟨location list⟩}

This is internally used by \glossaryentrynumbers. The nonumberlist option redefines \glossaryentrynumbers so that it doesn’t display the number list, but it still saves the number list in case it’s required. The desired font formatting for the location list can now more easily be set by redefining \GlsXtrFormatLocationList, without interfering with \glossaryentrynumbers.

If you want to suppress the number list always use the nonumberlist option instead of redefining \glossaryentrynumbers to do nothing.

Note that if you are using the "unsrt" family of commands the location list will only be present if the appropriate field has been set (see §8.4.2). There’s no need to save locations with bib2gls or with \printnoidxglossary because this is performed automatically (unlike \printglossary where the trick with \glossaryentrynumbers is required to capture the location list).

Sometimes users like to insert “page” or “pages” in front of the location list. This is quite fiddly to do with the base glossaries package, but glossaries-extra provides a way of doing this. First you need to enable this option and specify the text to display using:

\GlsXtrEnablePreLocationTag{⟨page tag⟩}{⟨pages tag⟩}

where ⟨page tag⟩ is the text to display if the location list only contains a single location and ⟨pages tag⟩ is the text to display otherwise. For example:
8. Defining and Displaying Glossaries

\GlsXtrEnablePreLocationTag{Page: }{Pages: }

An extra run is required when using this command.

Use \glsignore not \@gobble as the format if you want to suppress the page number (and only index the entry once).

See the accompanying sample file sample-pages.tex for an example.

Note that \bib2gls the \texttt{loc-prefix} resource option inserts a prefix at the start of non-empty location lists, which can be used as an alternative to \GlsXtrEnablePreLocationTag. There is also a corresponding \texttt{loc-suffix} option to provide a suffix.

Location lists displayed with \texttt{\printnoidxglossary} internally use: \glsxtrdisplayloc This command is provided by glossaries, but is modified by glossaries-extra to check for the start and end range formation identifiers ( and ) which are discarded to obtain the actual control sequence name that forms the location formatting command.

If the range identifiers aren’t present, this just uses

\begin{verbatim}
\glsxtrdisplaysingleloc{⟨format⟩}{⟨location⟩}
\end{verbatim}

otherwise it uses

\begin{verbatim}
\glsxtrdisplaystartloc{⟨format⟩}{⟨location⟩}
\end{verbatim}

for the start of a range (where the identifier has been stripped from ⟨format⟩) or

\begin{verbatim}
\glsxtrdisplayendloc{⟨format⟩}{⟨location⟩}
\end{verbatim}

for the end of a range (where the identifier has been stripped from ⟨format⟩).

By default the start range command saves the format in:

\begin{verbatim}
\glsxtrlocrangefmt
\end{verbatim}

and does:
8. Defining and Displaying Glossaries

(If the format is empty, it will be replaced with \texttt{glsnumberformat}.)

The end command checks that the format matches the start of the range, does:

\begin{Verbatim}
\glsxtrdisplayendlochook
\end{Verbatim}

(which does nothing by default), followed by:

\begin{Verbatim}
\glsxtrdisplaysingleloc{⟨format⟩}{⟨location⟩}
\end{Verbatim}

and then sets \texttt{\glsxtrlocrangefmt} to empty.

This means that the list

\begin{Verbatim}
\glsnoidxdisplayloc{}{page}{(textbf}{1},
\glsnoidxdisplayloc{}{page}{textbf}{1},
\glsnoidxdisplayloc{}{page}{})textbf}{1}.
\end{Verbatim}

doesn’t display any differently from

\begin{Verbatim}
\glsnoidxdisplayloc{}{page}{textbf}{1},
\glsnoidxdisplayloc{}{page}{textbf}{1},
\glsnoidxdisplayloc{}{page}{textbf}{1}.
\end{Verbatim}

but it does make it easier to define your own custom list handler that can accommodate the ranges.

8.6.4. Indexing Groups

The letter or symbol or number groups are a by-product of the indexing application. These are usually determined during the sorting according to the first (significant) character of the sort value. If the first character is an alphabetical character, the group is a letter group, with the group label the same as the letter. If the sort value is numeric, the group is a number group, with the label \texttt{glsnumbers}, otherwise the group is a symbol group with the label \texttt{glssymbols}.

For the “unsrt” family of commands, see §8.4.1 for more details about how group headers are inserted into the glossary. Only those commands are able to support sub-groups.
8. Defining and Displaying Glossaries

With \texttt{xindy}, the number group is automatically provided with the \texttt{xindy=glsnumbers} package option. It can be suppressed with \texttt{xindy={glsnumbers=false}} (see the base glossaries user manual for further details).

With \texttt{bib2gls}, group formation requires \texttt{--group} (or \texttt{-g}). This setting is off by default to allow for a faster process where no groups are required. When this setting is on, there are additional groups, depending on the sort method. For example, if you use a date-time sort method, then you will have date-time groups.

Take care not to confusion groups with hierarchy. See Gallery: Logical Glossary Divisions (type vs group vs parent)\textsuperscript{a} for the difference between the \texttt{group}, \texttt{type} and \texttt{parent} fields.

\textsuperscript{a}dickimaw-books.com/gallery/index.php?label=logicaldivisions

The base glossaries package provides a simplistic way of assigning a title to a group to allow for the use of language-sensitive commands \texttt{\textbackslash{glsymbolsgroupname}} and \texttt{\textbackslash{glsnumbers-groupname}}, which correspond to the \texttt{glssymbols} and \texttt{glsnumbers} groups. The more flexible groups that can be created with \texttt{bib2gls} require a better approach that is less likely to cause a conflict.

\begin{Verbatim}
\texttt{\textbackslash{glsxtrsetgrouptitle}{\langle group-label\rangle}{\langle group-title\rangle}}
\end{Verbatim}

Globally assigns the given title \texttt{\langle group-title\rangle} to the group identified by \texttt{\langle group-label\rangle}. This command is used implicitly within the \texttt{glstex} file to assign titles to groups obtained by \texttt{bib2gls}. Judicious definitions of the helper commands provided by \texttt{bib2gls} can provide a more flexible way of assigning groups.

\begin{Verbatim}
\texttt{\textbackslash{glsxtrlocalsetgrouptitle}{\langle group-label\rangle}{\langle group-title\rangle}}
\end{Verbatim}

As above but the assignment is local.

\begin{Verbatim}
\texttt{\textbackslash{glsxtrgetgrouptitle}{\langle group-label\rangle}{\langle cs\rangle}}
\end{Verbatim}

Obtains the title corresponding to the group identified by \texttt{\langle group-label\rangle} and stores the result in \texttt{\langle cs\rangle}. This command first checks if a title has been assigned by \texttt{\textbackslash{glsxtrgetgrouptitle}} and then, for compatibility with the base glossaries package, it will test for the existence of \texttt{\langle group-label\rangle groupname} if \texttt{\langle group-label\rangle} is \texttt{glssymbols} or \texttt{glsnumbers} or a single character. If no title is obtained from any of these tests, then the title will be assumed to be the same as the label.
8. Defining and Displaying Glossaries

The `\printnoidxglossary` command has a slightly different method, which uses the character code so it’s not suitable with UTF-8. In general, `\printnoidxglossary` is best avoided, where possible, and is inappropriate for locale-sensitive sorting.

8.6.5. glossaries-extra-stylemods

The glossaries-extra-stylemods package (more conveniently loaded through the glossaries-extra stylemods option) modifies some of the predefined styles that are provided with the glossaries package.

Any styles loaded after glossaries-extra-stylemods won’t be patched.

The stylemods option may be provided without a value, in which case all currently defined styles will be patched. Alternatively, you can supply a comma-separated list as the value, which indicates that, for each `⟨element⟩` in the list, the package glossary-`⟨element⟩` should be loaded and, if it’s a package provided with the base glossaries package, patched. For example:

```
\usepackage{glossaries-extra}
\usepackage{glossary-longragged}
\usepackage{glossary-mcols}
\usepackage{glossaries-extra-stylemods}
\setglossarystyle{mcolindex}
```

is equivalent to:

```
\usepackage[stylemods={longragged,mcols},style=mcolindex]{glossaries-extra}
```

You may prefer to combine `stylemods=with nostyles` to reduce the overhead of loading unnecessary packages.

The glossaries-extra-stylemods package adjusts the predefined styles so that they all use `\glspostdescription` and replaces any hard-coded space before the location list with

```
\glsxtrprelocation initial: \space
```

You can therefore redefine that command in combination with `postpunc` to alter the separator before the location list. For example, to have a comma followed by `\hfil`:
8. Defining and Displaying Glossaries

\usepackage[postpunc=comma,stylemods]{glossaries-extra}
\renewcommand{\glsxtrprelocation}{\hfil}

Be careful with doing this as it will look odd if the location list is missing.

With bib2gls you can instead redefine \glsxtrprelocation to do nothing and set the location prefixes with loc-prefix which will only apply if the entry has a location list. Alternatively, you could redefine \glsxtrprelocation to check if the location field is set.

8.6.5.1. Inline Style

The patched inline style is dealt with slightly differently. The original definition provided by the glossary-inline package uses \glspostdescription at the end of the glossary (not after each entry description) within the definition of \glspostinline. The style modification changes this so that \glspostinline just does a full stop followed by space factor adjustment, and the description \glsinlinedescformat and sub-entry description formats \glsinlinesubdescformat are redefined to include \glsxtrpostdescription (not \gls-postdescription). This means that the modified inline style isn’t affected by the nopostdot option, but the category post-description hook can still be used.

8.6.5.2. Tabular Styles

The tabular-like styles, such as long are adjusted so that the \ifglsnogroupskip conditional (set with nogroupskip) is moved outside of the definition of \glsgroupskip to avoid problems that cause an “Incomplete \iftrue” error with \printunsrtglossary and \printnoidxglossary. This means that if you want to change this conditional using \setupglossaries or using the nogroupskip option in \printglossary, \printnoidxglossary or \printunsrtglossary, you must also reset the glossary style.

8.6.5.3. List Styles

The list styles use:

\glslistprelocation initial: \glsxtrprelocation

(which defaults to \glsxtrprelocation) for top-level items and:

\glslistchildprelocation initial: \glslistprelocation
8. Defining and Displaying Glossaries

(which defaults to \glstprelocation) for child items.

The description (including the post-description hook) is governed by:

\glstdes{⟨entry-label⟩}

for the list and altlist styles (but not the listdotted variations).

The hard-coded \item[⟨target and name⟩] is replaced with:

\glstitem{⟨entry-label⟩}

The altlist styles use:

\glstaltlistitem{⟨entry-label⟩}

which internally uses \glstitem. The header item (for the list styles that should the group title, such as listgroup) is governed by:

\glstgroupheaderitem{⟨group-label⟩}{⟨header code⟩}

This ignores the ⟨group-label⟩ by default and simply places the second argument in the optional argument of \item. The ⟨header code⟩ is the formatted group title, possibly including a hypertarget. The spacing after the group item is given by:

\glstgroupafterheader

For just the list style and its letter group variations (not the altlist or listdotted variations) the location list for child entries is followed by:

\glstchildpostlocation

which defaults to a full stop.

The default value of \glstdwidth is changed so that it’s set at the start of the document (if it hasn’t been changed in the preamble). This should take into account situations where \hsize isn’t set until the start of the document.
The separator between groups (if not nogroupskip) is now given by:

\glslistgroupskip

This defaults to \indexspace with penalties to deter page breaks. This command isn’t used if nogroupskip is set.

### 8.6.5.4. Tree Styles

The group headings for styles like treegroup are formatted with:

\glstreegroupheaderfmt\{⟨text⟩\}

The navigation elements for styles like treehypergroup is formatted with:

\glstreenavigationfmt\{⟨text⟩\}

The above two commands are defined in terms of \glstreenamefmt, since that was the command originally used for the group headings and navigation. This now allows these different elements to be defined independently, but the most common redefinition is for \glstreenamefmt to remove the bold in the name. If the bold is still required for the group heading and navigation elements, then both other commands also need redefining. To simplify matters, all three commands have been defined to use:

\glstreedefaultnamefmt\{⟨text⟩\}

This simply does \textbf{⟨text⟩}.

This means that if you want to change all three to use a particular style you only need to redefine \glstreedefaultnamefmt, but if you only want to redefine \glstreenamefmt without affecting the other two commands, then you now can.

The separator between groups without headers is given by:

\glstreegroupskip

This defaults to just \indexspace without penalties. This command isn’t used if nogroupskip is set. (The penalties introduced in v1.41 were moved to \glstreegroupheaderskip in v1.42 as they are inappropriate when there’s no header.)
The separator between groups with headers is now given by:
\glstreegroupheaderskip

This defaults to \glstreegroupskip with penalties to deter page breaks after the group heading.
The styles that display the group titles now use:
\glstreePreHeader\{⟨group-label⟩\}⟨⟨group-title⟩⟩

This does nothing by default and is inserted before the group title. You can redefine it to add the group title to the PDF bookmarks. For example, if the glossary title uses \chapter then:
\renewcommand\{\glstreePreHeader\}[2]{% 
  \pdfbookmark[1]{#2}{\currentglossary.#1}%
}\}

will insert section-level bookmarks. The use of \currentglossary helps to provide unique bookmark labels in the event of multiple glossaries.

The glossary–tree package provides the commands
\glstreepredesc
and
\glstreechildpredesc
(which both default to a space) and uses them in the tree-like styles, but not for the alttree style. The glossaries–extra–stylemods package modifies the alttree style so that it has equivalent hooks:
\glsalttreepredesc
and
\glsalttreechildpredesc
8. Defining and Displaying Glossaries

These do nothing by default.

The index-like and tree-like styles insert the pre-location list space with:

\glstreeprelocation \textit{initial:} \glxsxtrprelocation

(which defaults to \glxsxtrprelocation) for top-level items and

\glstreechildprelocation \textit{initial:} \glstreeprelocation

(which defaults to \glstreeprelocation) for child items.

The styles like treenoname use:

\glstreenonamedesc{⟨entry-label⟩}

to display the pre-description separator, the description and the post-description hook. Similarly for the symbol:

\glstreenamesymbol{⟨entry-label⟩}

The above are just used for top-level entries. Child entries don’t have the name or symbol displayed for the treenoname styles, so there’s only a command for the child description:

\glstreenamechilddesc{⟨entry-label⟩}

For the tree styles (but not the treenoname or alttree styles), the description is displayed using:

\glstreedesc{⟨entry-label⟩}

and the symbol with:

\glstreesymbol{⟨entry-label⟩}
Again the above two commands are just for top-level entries. The child entries use:

\glstreechilddesc{⟨entry-label⟩}

for the description and

\glstreechildsymbol{⟨entry-label⟩}

for the symbol. There are now wrapper commands for \glstreedesc and \glstreechilddesc that check for the description and symbol to determine what separator to use before the page list:

\glstreeDescLoc{⟨entry-label⟩}{⟨location list⟩}

for top-level entries and

\glstreeChildDescLoc{⟨entry-label⟩}{⟨location list⟩}

for sub-entries.

If either the symbol or description is present these will use \glstreeprelocation or \glstreechildprelocation, respectively. Otherwise, both will use:

\glstreeNoDescSymbolPreLocation

The default is a space. This means that you could have, say, a comma followed by a space for terms that are simply an alias, but just have a space for terms that have a description that ends with a full stop (or that just have a symbol without a description) where the comma would be inappropriate.

Version 1.42 has corrected an error that was introduced to v1.41 that caused the name to run into the location list if there was no symbol and no description.

There are some additional commands for use with the alttree style to make it easier to modify. These commands are only defined if the glossary–tree package has already been loaded, which is typically the case unless the notree or nostyles option has been used.
when loading glossaries.

\gglssetwidest[⟨level⟩]{⟨name⟩}

This is like \glssetwidest but performs a global assignment.

\eglssetwidest[⟨level⟩]{⟨name⟩}

This is like \glssetwidest but expands ⟨name⟩.

\xglssetwidest[⟨level⟩]{⟨name⟩}

This is like \eglssetwidest but performs a global assignment.

The following only set the value if ⟨name⟩ is wider than the current value. Local update:

\glsupdatewidest[⟨level⟩]{⟨name⟩}

Global update:

\gglsupdatewidest[⟨level⟩]{⟨name⟩}

Locale update (expands ⟨name⟩):

\eglsupdatewidest[⟨level⟩]{⟨name⟩}

Global update (expands ⟨name⟩):

\xglsupdatewidest[⟨level⟩]{⟨name⟩}

The widest entry value can later be retrieved using:

\glsgetwidestname
which expands to the widest top-level name and:

\glsetwidestsubname{⟨level⟩}

expands to either the widest name for the given hierarchical level or to the widest top-level name, if no widest name set for ⟨level⟩.

Note that if you are using bib2gls, you can use the resource option set-widest which will try to determine the widest name of all the selected entries. This isn’t guaranteed to work as it may depend on fonts or commands that bib2gls can’t replicate, but it should be suitable for names that just consist of text, and can be more efficient than iterating over all the defined entries using \TeX.

The command \glsfindwidesttoplevelname provided by glossary-tree has a Camel-Case synonym:

\glsFindWidestTopLevelName

Similar commands are also provided. If the optional ⟨glossary labels⟩ is omitted, the list of all non-ignored glossaries is assumed.

\glsFindWidestUsedTopLevelName[⟨glossary labels⟩]

This has an additional check that the entry has been used. Naturally this is only useful if the glossaries that use the alttree style occur at the end of the document. This command should be placed just before the start of the glossary. (Alternatively, place it at the end of the document and save the value in the auxiliary file for the next run.)

\glsFindWidestUsedAnyName[⟨glossary labels⟩]

This is like the previous command but if doesn’t check the parent key. This is useful if all hierarchical levels should have the same width for the name.

\glsFindWidestAnyName[⟨glossary labels⟩]

This is like the previous command but doesn’t check if the entry has been used.
8. Defining and Displaying Glossaries

This is like \glsFindWidestUsedTopLevelName but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

\glsFindWidestLevelTwo[⟨glossary labels⟩]

This is like the previous command but doesn’t check if the entry has been used.

\glsFindWidestUsedAnyNameSymbol[⟨glossary labels⟩]{⟨register⟩}

This is like \glsFindWidestUsedAnyName but also measures the symbol. The length of the widest symbol is stored in ⟨register⟩.

\glsFindWidestAnyNameSymbol[⟨glossary labels⟩]{⟨register⟩}

This is like the previous command but it doesn’t check if the entry has been used.

\glsFindWidestUsedAnyNameSymbolLocation[⟨glossary labels⟩]{⟨register1⟩}{⟨register2⟩}

This is like \glsFindWidestUsedAnyNameSymbol but also measures the location list. This requires \glsentrynumberlist. The length of the widest symbol is stored in ⟨register1⟩ and the length of the widest location list is stored in ⟨register2⟩.

\glsFindWidestAnyNameSymbolLocation[⟨glossary labels⟩]{⟨register1⟩}{⟨register2⟩}

This is like the previous command but it doesn’t check if the entry has been used.

\glsFindWidestUsedAnyNameLocation[⟨glossary labels⟩]{⟨register⟩}

This is like \glsFindWidestUsedAnyNameSymbolLocation but doesn’t measure the symbol. The length of the widest location list is stored in ⟨register⟩.

\glsFindWidestAnyNameLocation[⟨glossary labels⟩]{⟨register⟩}

This is like the previous command but doesn’t check if the entry has been used.
8. Defining and Displaying Glossaries

The layout of the symbol, description and location list is governed by:

\begin{verbatim}
\glsxtralttreeSymbolDescLocation\{\textit{entry-label}\}\{\textit{location list}\}
\end{verbatim}

for top-level entries and

\begin{verbatim}
\glsxtralttreeSubSymbolDescLocation\{\textit{entry-label}\}\{\textit{location list}\}
\end{verbatim}

for sub-entries.

There is now a user level command that performs the initialisation for the alttree style:

\begin{verbatim}
\glsxtralttreeInit
\end{verbatim}

The paragraph indent for subsequent paragraphs in multi-paragraph descriptions is provided by the length:

\begin{verbatim}
\glsxtrAltTreeIndent
\end{verbatim}

For additional commands that are available with the alttree style, see the documented code (glossaries-extra-code.pdf). See also the accompanying sample files sample-alttree.tex, sample-alttree-sym.tex and sample-alttree-marginpar.tex.

8.7. New Glossary Styles

The glossaries-extra package comes with some new styles. The associated style package needs to be loaded. This can be done with \texttt{\usepackage} but it’s simpler to use the \texttt{stylemods} option. For example:

\begin{verbatim}
\usepackage[stylemods=bookindex,style=bookindex]{glossaries-extra}
\end{verbatim}

If you don’t require any of the base styles, use \texttt{nostyles} (but note that some style packages automatically load another style package if it the style builds on an existing one).

8.7.1. glossary-bookindex package

The glossary-bookindex package provides the glossary style bookindex. This is very similar to the mcolindexgroup style but is designed for indexes, so by default only the name and location list are displayed. This style is demonstrated in Example 146 on the following page (using \texttt{bib2gls}). Note that some entries don’t have location lists because they weren’t
recorded in the document, but were included as dependencies. See §5.9.1 for dealing with cross-references that may not be required.

Example 146: The bookindex style

Glossary

A
adipiscing, 1
alias-lorem, see lorem
amet, 1, see dolor
arcu, 1–3, see also placerat, vitae & curabitur
augue, 2

B
consectetuer, 2
vulputate, 1

C
curabitur

D
dolor
donec, 2

e, see vulputate

E
et, see vulputate

F

G

H

I

J

K

L
lectus
urna, see nulla

M
mauris

N
nonummy, see also mauris
consectetuer, 2
vulputate, 1
nulla, 3

P
pellentesque, see augue
placert, 1

Q

R

S

T

U

V
vitae

The bookindex style only supports a maximum hierarchical level of 2 (top-level, level 1 and level 2). It’s primarily designed for use with bib2gls. It may be used with other indexing options, but some features may not be present and UTF-8 characters may cause a problem with non-Unicode engines in letter group headings or PDF bookmarks. (bib2gls uses numeric identifies by default to avoid these problems, see §8.4.1.)

The number of columns is given by:

\glsxtrbookindexcols initial: 2

which defaults to 2.
8. Defining and Displaying Glossaries

This style uses the multicols environment. If the command:

\glsxtrbookindexcolspread

isn’t empty then it’s supplied as the optional argument following \begin{multicols}{⟨n⟩}. You can switch from multicols to multicols* by redefining:

\glsxtrbookindexmulticolsenv

For example:

\renewcommand{\glsxtrbookindexmulticolsenv}{multicols*}

Each top-level entry is displayed using:

\glsxtrbookindexname{⟨entry-label⟩}

This just does \glossentryname{⟨entry-label⟩} by default. For example, if you want the symbol to be included:

\renewcommand*{\glsxtrbookindexname}[1]{%
  \glossentryname{#1}%
  \ifglshassymbol{#1}{\space (\glossentrysymbol{#1})}%
}%

or if you want the description (if set):

\renewcommand*{\glsxtrbookindexname}[1]{%
  \glossentryname{#1}%
  \ifglshasdesc{#1}{\space \glossentrydesc{#1}\glspostdescription}%
}%

(which picks up the post-description hook).

Alternatively you can use the \glsxtrpostname\langle\textit{category}\rangle hook to append information after the name according to the entry’s category.
Sub-entries are displayed using:

\glsxtrbookindexsubname{(entry-label)}

which just defaults to \glsxtrbookindexname{(entry-label)}.

The separator used before the location list for top-level entries is given by:

\glsxtrbookindexprelocation{(entry-label)}

where \textit{(entry-label)} is the entry's label. This checks if the location field has been set. If it has, it does:

,\glsxtrprelocation

otherwise it just does \glsxtrprelocation (which defaults to \texttt{space}) with no comma. If you’re using \texttt{bib2gls} with save-locations=false, the location field won’t be set.

The separator used before the location list for sub-entries is given by:

\glsxtrbookindexsubprelocation{(entry-label)}

which defaults to \glsxtrbookindexprelocation{(entry-label)}.

The actual location list is encapsulated with:

\glsxtrbookindexlocation{(entry-label)}{(location list)}

for top-level entries and:

\glsxtrbookindexsublocation{(entry-label)}{(location list)}

for sub-entries. These both just do \textit{(location list)} by default.

The separator used between a top-level parent and child entry is given by:

\glsxtrbookindexparentchildsep

This defaults to \texttt{\nopagebreak}. 

440
The separator used between a sub-level parent and child entry is given by:
\glsxtrbookindexparentsubchildsep

This defaults to \glsxtrbookindexparentchildsep.
The separator between top-level entries is given by:
\glsxtrbookindexbetween{⟨entry1-label⟩}{⟨entry2-label⟩}

This comes after the entry given by ⟨entry1-label⟩, if the entry has no children, or after the last descendent otherwise, so it always comes immediately before the entry given by ⟨entry2-label⟩ unless the entry occurs at the start of a group. This does nothing by default.

The separator between two level 1 entries is given by:
\glsxtrbookindexsubbetween{⟨entry1-label⟩}{⟨entry2-label⟩}

The separator between two level 2 entries is given by:
\glsxtrbookindexsubsubbetween{⟨entry1-label⟩}{⟨entry2-label⟩}

At the end of each letter group, the following hooks are done in order:
\glsxtrbookindexsubsubatendgroup{⟨entry-label⟩}

where ⟨entry-label⟩ is the label of the last level 2 entry
\glsxtrbookindexsubatendgroup{⟨entry-label⟩}

where ⟨entry-label⟩ is the label of the last level 1 entry
\glsxtrbookindexatendgroup{⟨entry-label⟩}

where ⟨entry-label⟩ is the label of the last level 0 entry.

For example, the resource option seealso=omit instructs bib2gls to omit the seealso cross-reference from the location list. (The see cross-reference will still be added unless you also have see=omit.) The seealso cross-reference can instead be appended after the child entries using:
This uses \glistreesubitem and \glistreesubsubitem to indent the cross-reference according to the next level down, so the cross-reference for a top-level entry is aligned with the sub-entries, and a level 1 entry has its cross-reference aligned with sub-sub-entries. In the event that a level 2 entry has a cross-reference, this is indented a bit further (but it won’t be aligned with any deeper level as the bookindex style only supports a maximum of two sub-levels).

The bookindex style uses group headings. (If you use bib2gls remember to invoke it with the --group or -g switch, see §8.4.1.) The heading will use:

\glsxtrbookindexbookmark\{⟨group-title⟩\}\{⟨bookmark-name⟩\}

If \pdfbookmark has been defined, this will use that command to bookmark the group title. If section=chapter is set (default if chapters are defined) then this uses level 1 otherwise it uses level 2. You can redefine this command if this isn’t appropriate. If \pdfbookmark hasn’t been defined, this command does nothing.
The group heading is formatted according to:

\glsxtrbookindexformatheader{⟨group-title⟩}

which is defined as:

\newcommand*{\glsxtrbookindexformatheader}[1]{\par{\centering\glstreegroupheaderfmt{#1}\par}}

where \glstreegroupheaderfmt is provided by the glossary-tree package, which is automatically loaded. Note that the entry names aren’t encapsulated with \glstreenamefmt.

The skip after a group header is given by:

\glsxtrbookindexpregroupskip{⟨skip⟩}

The argument is the skip that would normally be inserted if there wasn’t a group header.

The glossary-bookindex package provides some supplementary commands that aren’t used by default, but may be used when adjusting the style. These commands should only be used within one of the \print...glossary commands. (That is, they should only be used in glossary styles or in hooks.)

\glsxtrbookindexmarkentry{⟨entry-label⟩}

This writes information to the aux file that can be read on the next run to obtain the first and last entry on each page of the glossary.

You can display the first entry associated with the current page using:

\glsxtrbookindexfirstmark{⟨entry-label⟩}

and the last entry associated with the current page using:

\glsxtrbookindexlastmark{⟨entry-label⟩}

These do nothing if there are no entries marked on the current page (or if the document build isn’t up to date).
8. Defining and Displaying Glossaries

The entry is formatted using:

\glsxtrbookindexfirstmarkfmt{\langle entry-label\rangle}

for the first instance and

\glsxtrbookindexlastmarkfmt{\langle entry-label\rangle}

for the last.

These commands are designed for use in page headers or footers where the page number is stable. For example, \glsxtrbookindexname can be redefined to mark the current entry:

\renewcommand{\glsxtrbookindexname}[1]{\glsxtrbookindexmarkentry{#1}\glossentryname{#1}}

If you only want to mark the top-level entries, remember to redefine \glsxtrbookindexsubname as it defaults to \glsxtrbookindexname:

\renewcommand{\glsxtrbookindexsubname}[1]{\glossentryname{#1}}

Then if you’re using fancyhdr you can set the page style to show the first and last entry for the current page with:

\pagestyle{fancy}%
\lhead{\thepage}%
\lfoot{\glsxtrbookindexfirstmark}%
\cfoot{\%}
\rfoot{\glsxtrbookindexlastmark}\

8.7.2. glossary–longextra package

The glossary–longextra package provides additional tabular-like styles similar to those provided by glossary–longbooktabs (which is automatically loaded). These don’t support hierarchical
levels except for homographs (level 1 entries with the same name as their parent).

By default, these styles use the longtable environment, but if you know that your glossary won’t span more than a page and you need to use it in a context that’s incompatible with longtable, you can instead setup these styles to use tabular instead. In order to do this you must use:

\GlsLongExtraUseTabulartrue

*before the style is set.* If you later want to switch back to using longtable for another glossary, use:

\GlsLongExtraUseTabularfalse

(or scope \GlsLongExtraUseTabulartrue). Again, the style must be set after this change to the conditional is implemented. You can test this setting with:

\ifGlsLongExtraUseTabulartrue\elsefalse\fi

initial: \iffalse

For example:

\GlsLongExtraUseTabulartrue
\setglossarystyle{long-name-desc}

or

\GlsLongExtraUseTabulartrue
\printunsrtglossary[style=long-name-desc]

If you switch to tabular, the default vertical alignment is obtained from:

\glslongextraTabularVAlign

initial: c

This should expand to one of: c (centred), t (top) or b (bottom). The default is c.

For either tabular or longtable, the column titles are formatted according to:

\glslongextraHeaderFmt{⟨text⟩}
which simply does $\textbf{⟨text⟩}$ by default. As with the long-like styles, the header text for the columns are given by the language-sensitive commands: \entryname, \descriptionname, \symbolname and \pagelistname.

Most styles show the name which, as with other predefined styles, also includes the entry item number (if entrycounter is on) and hypertarget anchor. These are all performed for top-level entries with:

\begin{verbatim}
\glslongextraNameFmt{⟨entry-label⟩}
\end{verbatim}

This uses \glossentryname, so it supports the post-name hook and associated attributes. Child entries are displayed with:

\begin{verbatim}
\glslongextraSubNameFmt{⟨level⟩}{⟨entry-label⟩}
\end{verbatim}

This includes the sub-entry item number (if subentrycounter is on) and the hypertarget anchor. The actual name isn’t shown by default.

The horizontal alignment for the name column is obtained with:

\begin{verbatim}
\glslongextraNameAlign
\end{verbatim}

This expands to 1 by default.

For styles that show the description, that’s formatted with:

\begin{verbatim}
\glslongextraDescFmt{⟨entry-label⟩}
\end{verbatim}

for top-level entries, which uses \glossentrydesc and the post-description hook, and

\begin{verbatim}
\glslongextraSubDescFmt{⟨level⟩}{⟨entry-label⟩}
\end{verbatim}

for child entries (which just uses \glslongextraDescFmt).

The horizontal alignment for the description column is obtained with:

\begin{verbatim}
\glslongextraDescAlign
\end{verbatim}

This expands to >{\raggedright}p{\glssdescwidth} by default. This means ragged-right paragraph style with width given by \glssdescwidth. (See the documentation for the array package for information about this alignment syntax.) If a widest name has been set, \glssdescwidth will be calculated according to the best fit for the given style.
If you are using bib2gls, you may be able to use the `set-widest` option, otherwise to set the widest name, use:

```
\glslongextraSetWidest{⟨widest-name⟩}
```

If you have already used \glssetwidest provided with the alttree style, the default widest name will be obtained from that, but note that only level 0 is supported for the glossary -longextra styles.

You can update the widest name with:

```
\glslongextraUpdateWidest{⟨name⟩}
```

This is like \glslongextraSetWidest but will only set the new value if it’s wider than the current widest name.

Although these styles don’t support hierarchy, the following is provided for child entries:

```
\glslongextraUpdateWidestChild{⟨level⟩}{⟨name⟩}
```

This does nothing by default. If \glslongextraSubNameFmt is redefined to show the child name, then the above command will need to be redefined to use \glslongextraUpdateWidest.

For styles that show the location list, that’s formatted with:

```
\glslongextraLocationFmt{⟨entry-label⟩}{⟨location list⟩}
```

for top-level entries. Child location lists are formatted with:

```
\glslongextraSubLocationFmt{⟨level⟩}{⟨entry-label⟩}{⟨location list⟩}
```

Both of these simply do the ⟨location list⟩ argument.

The horizontal alignment for the location list column is obtained with:

```
\glslongextraLocationAlign
```

This expands to >\{\raggedright\}p{\glspagelistwidth} by default. This means ragged-right paragraph style with width given by \glspagelistwidth.

447
8. Defining and Displaying Glossaries

For styles that show the symbol (in addition to the name), that’s formatted with:

\texttt{\textbackslash glslongextraSymbolFmt\{⟨entry-label⟩\}}

for top-level entries. This simply uses \texttt{\textbackslash glossentrysymbol}. Child entries use:

\texttt{\textbackslash glslongextraSubSymbolFmt\{(level)\}\{⟨entry-label⟩\}}

which uses \texttt{\textbackslash glslongextraSymbolFmt}.

The horizontal alignment for the symbol column (except for the long-sym-desc and long-desc-sym styles) is obtained with:

\texttt{\textbackslash glslongextraSymbolAlign initial: c}

This expands to \texttt{c} by default.

Top-level group headings are formatted with:

\texttt{\textbackslash glslongextraGroupHeading\{(number columns)\}\{(group-label)\}}

The first argument is the total number of columns in the table. For example, 2 for the long-name-desc style or 3 for the long-name-sym-desc style. The second argument is the group’s label (not the title). This command does nothing by default. (If you are using \texttt{bib2gls}, remember that you need to use the --group or -g switch to support groups.)

Sub-level groups are only supported with the “unsrt” family of commands (see §8.4.1). When they are supported, the heading will be formatted with:

\texttt{\textbackslash glslongextraSubGroupHeading\{(number columns)\}\{(prev group level)\}\{(group level)\}\{(parent-entry-label)\}\{(group-label)\}}

The styles are sub-divided below into the set of elements that are shown in each column, which may consist of: \texttt{name, symbol, description} or location list. There will be blank cells if any of the corresponding fields have not been set or if the location list has been suppressed.
8. Defining and Displaying Glossaries

8.7.2.1. Name and Description Only

These styles don’t display the symbol or location list, regardless of whether or not they have been set. In each case, the style starts with:

\texttt{\glslongextraSetDescWidth}

which updates \glsdescwidth according to the widest name, identified with \glslongextraSetWidest. The column header text is also taken into account. If a widest name hasn’t been set and the column header is shorter than one or more names, the description column may be too wide. The value of \glsdescwidth is calculated as \texttt{\linewidth - 4\tabcolsep - W}, where \texttt{W} is the width of the widest name.

If you want to set \glsdescwidth to a specific value, then redefine \texttt{\glslongextraSetDescWidth} with the desired length assignment.

\texttt{\glslongextraNameDescTabularHeader}

This has two columns: the name on the left and the description on the right. The table header is given by:

\texttt{\glslongextraNameDescTabularFooter}

which shows the column headers with horizontal rules. The table footer is given by:

\texttt{\glslongextraNameDescHeader}

which just does a horizontal rule. With longtable, the table header and footer are set with:

\texttt{\glslongextraNameDescHeader}

which uses the above header and footer commands.

\texttt{\glslongextraNameDescFooter}

This has two columns: the name on the right and the description on the left. The table header
is given by:

```
\glslongextraDescNameTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
\glslongextraDescNameTabularFooter
```

which just does a horizontal rule. With `longtable`, the table header and footer are set with:

```
\glslongextraDescNameHeader
```

which uses the above header and footer commands.

### 8.7.2.2. Name, Symbol and Description Only

These styles don’t show the location list. In each case, the style starts with:

```
\glslongextraSymSetDescWidth
```

which updates `\glsdescwidth` according to the widest name, identified with `\glslongextraSetWidest`. This starts by calculating `\glsdescwidth` with `\glslongextraSetDescWidth` and then subtracts the width of the symbol column header text (which is assumed to be the widest text in that column).

If you want to set `\glsdescwidth` to a specific value, then redefine `\glslongextraSymSetDescWidth` with the desired length assignment.

```
long-name-desc-sym
```

This has three columns: the name on the left, the description in the middle and the symbol on the right. The table header is given by:

```
\glslongextraNameDescSymTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
\glslongextraNameDescSymTabularFooter
```
which just does a horizontal rule. With longtable, the table header and footer are set with:

```
\glslongextraNameDescSymHeader
```

which uses the above header and footer commands.

```
long-name-sym-desc
```

This has three columns: the name on the left, the symbol in the middle and the description on the right. The table header is given by:

```
\glslongextraNameSymDescTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
\glslongextraNameSymDescTabularFooter
```

which just does a horizontal rule. With longtable, the table header and footer are set with:

```
\glslongextraNameSymDescHeader
```

which uses the above header and footer commands.

```
long-sym-desc-name
```

This has three columns: the name on the right, the description in the middle and the symbol on the left. The table header is given by:

```
\glslongextraSymDescNameTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
\glslongextraSymDescNameTabularFooter
```
8. Defining and Displaying Glossaries

which just does a horizontal rule. With longtable, the table header and footer are set with:

\glslongextraSymDescNameHeader

which uses the above header and footer commands.

long-desc-sym-name

This has three columns: the name on the right, the symbol in the middle and the description on the left. The table header is given by:

\glslongextraDescSymNameTabularHeader

which shows the column headers with horizontal rules. The table footer is given by:

\glslongextraDescSymNameTabularFooter

which just does a horizontal rule. With longtable, the table header and footer are set with:

\glslongextraDescSymNameHeader

which uses the above header and footer commands.

8.7.2.3. Name, Description and Location Only

These styles don’t display the symbol, regardless of whether or not the symbol field has been set. In each case, the style starts with:

\glslongextraLocSetDescWidth

which updates \glsdescwidth according to the widest name, identified with \glslongextraSetWidest. This starts by calculating \glsdescwidth with \glslongextraSetDescWidth and then subtracts 2)tabcolsep - \glspagelistwidth.

If you want to set \glsdescwidth to a specific value, then redefine \glslongextraLocSetDescWidth with the desired length assignment.

long-name-desc-loc
This has three columns: the name on the left, the description in the middle and the location list on the right. The table header is given by:

```
glslongextraNameDescLocationTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
glslongextraNameDescLocationTabularFooter
```

which just does a horizontal rule. With longtable, the table header and footer are set with:

```
glslongextraNameDescLocationHeader
```

which uses the above header and footer commands.

```
long-loc-desc-name
```

This has three columns: the name on the right, the description in the middle and the location list on the left. The table header is given by:

```
glslongextraLocationDescNameTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
glslongextraLocationDescNameTabularFooter
```

which just does a horizontal rule. With longtable, the table header and footer are set with:

```
glslongextraLocationDescNameHeader
```

which uses the above header and footer commands.
8. Defining and Displaying Glossaries

8.7.2.4. Name, Description, Symbol and Location

These styles show the name, description, symbol and location list. In each case, the style starts with:

\glslongextraSymLocSetDescWidth

which updates \glsdescwidth according to the widest name, identified with \glslongextraSetWidest. This starts by calculating \glsdescwidth with \glslongextraSymSetDescWidth and then subtracts 2*\tabcolsep – \glspagelistwidth.

If you want to set \glsdescwidth to a specific value, then redefine \glslongextraSymLocSetDescWidth with the desired length assignment.

long-name-desc-sym-loc

This has four columns, from left to right: the name, description, symbol and the location list. The table header is given by:

\glslongextraNameDescSymLocationTabularHeader

which shows the column headers with horizontal rules. The table footer is given by:

\glslongextraNameDescSymLocationTabularFooter

which just does a horizontal rule. With longtable, the table header and footer are set with:

\glslongextraNameDescSymLocationHeader

which uses the above header and footer commands.

long-name-sym-desc-loc

This has four columns, from left to right: the name, symbol, description and the location list. The table header is given by:

\glslongextraNameSymDescLocationTabularHeader
8. Defining and Displaying Glossaries

which shows the column headers with horizontal rules. The table footer is given by:

```
glslongextraNameSymDescLocationTabularFooter
```

which just does a horizontal rule. With `longtable`, the table header and footer are set with:

```
glslongextraNameSymDescLocationHeader
```

which uses the above header and footer commands.

```
long-loc-sym-desc-name
```

This has four columns, from left to right: the location list, symbol, description and the name. The table header is given by:

```
glslongextraLocationSymDescNameTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
glslongextraLocationSymDescNameTabularFooter
```

which just does a horizontal rule. With `longtable`, the table header and footer are set with:

```
glslongextraLocationSymDescNameHeader
```

which uses the above header and footer commands.

```
long-loc-desc-sym-name
```

This has four columns, from left to right: the location list, description, symbol and the name. The table header is given by:

```
glslongextraLocationDescSymNameTabularHeader
```
8. Defining and Displaying Glossaries

which shows the column headers with horizontal rules. The table footer is given by:

\glslongextraLocationDescSymNameTabularFooter

which just does a horizontal rule. With longtable, the table header and footer are set with:

\glslongextraLocationDescSymNameHeader

which uses the above header and footer commands.

8.7.2.5. Symbol and Description Only

These are two-column styles designed to show only the symbol and description. However, if the symbol isn’t set then the name will be used instead. If this occurs, you may need to change the width of the description column.

The horizontal alignment for the symbol column is obtained with:

\glslongextraSymbolNameAlign\textit{initial: 1}

which expands to 1 by default. Note that this is different from the alignment used for styles like long-name-sym-desc.

These styles have the entry item number (if entrycounter is on) and the hypertarget anchor (if enabled) in the symbol column since there’s no name shown (unless the symbol is missing). These are all performed by for top-level entries by:

\glslongextraSymbolTargetFmt\{⟨entry-label⟩\}

The symbol is formatted according to \glslongextraSymbolFmt. Child entries use:

\glslongextraSubSymbolTargetFmt\{⟨level⟩\}{⟨entry-label⟩}

Unlike \glslongextraSubNameFmt this shows the field value (formatted with \glslongextraSymbolFmt).

The following commands use the above if the symbol field is set, otherwise they show the name.

\glslongextraSymbolOrName\{⟨entry-label⟩\}
8. Defining and Displaying Glossaries

Shows the symbol, if set, or the name otherwise, with the target. Child entries use:

\glslongextraSubSymbolOrName{⟨level⟩}{⟨entry-label⟩}

Shows the symbol with \glslongextraSubSymbolTargetFmt, if set, or the name otherwise, with the target.

In each case, the style starts with:

\glslongextraSymNoNameSetDescWidth

which calculates \glsdescwidth as \linewidth − 4\tabcolsep − W, where W is the width of the symbol column header. Note that this assumes the content of the symbol column isn’t wider than the column header.

If you want to set \glsdescwidth to a specific value, then redefine \glslongextraSymNoNameSetDescWidth with the desired length assignment.

long-sym-desc

The symbol is in the left column (or the name, if the symbol isn’t set). The description is in the right. The location list isn’t shown. The table header is given by:

\glslongextraSymDescTabularHeader

which shows the column headers with horizontal rules. The table footer is given by:

\glslongextraSymDescTabularFooter

which just does a horizontal rule. With longtable, the table header and footer are set with:

\glslongextraSymDescHeader

which uses the above header and footer commands.

long-desc-sym

The symbol is in the right column (or the name, if the symbol isn’t set). The description is in
8. Defining and Displaying Glossaries

the left. The location list isn’t shown. The table header is given by:

```
\glslongextraDescSymTabularHeader
```

which shows the column headers with horizontal rules. The table footer is given by:

```
\glslongextraDescSymTabularFooter
```

which just does a horizontal rule. With longtable, the table header and footer are set with:

```
\glslongextraDescSymHeader
```

which uses the above header and footer commands.

8.7.2.6. Abbreviations Only

These styles are designed for abbreviations. They display the short and long forms, rather than the name and description, although these may happen to match. They are primarily intended for mini-glossaries or similar summary lists.

Although these styles don’t show the name or description, they still use some of the name and description settings provided by glossary-longextra. The column for the short form uses the same alignment as for the name columns (\glslongextraNameAlign). The column for the long form uses the same alignment as for the description columns (\glslongextraDescAlign) and has the width set to \glsdescwidth. However, the name and description formatting commands or attributes (such as \glsnamefont, glossnamefont or glossname) aren’t used as the formatting is left to the abbreviation style.

If the short field hasn’t been set, the short column will show the name instead, and if the long field hasn’t been set, the long column will show the description instead (using the same commands as for styles like long-name-desc, which do use the associated formatting commands and attributes).

These styles use the following commands:

```
\glslongextraShortHeader initial: \entryname
```

The header for the column showing the short form. This is defined as:

```
\newcommand{\glslongextraShortHeader}{\entryname}
```
8. Defining and Displaying Glossaries

\glslongextraLongHeader \textit{initial:} \texttt{\\descriptionname}

The header for the column showing the long form. This is defined as:

\newcommand{\glslongextraLongHeader}{\descriptionname}

\glslongextraShortTargetFmt{⟨entry-label⟩}

This governs the way that the short form should be displayed, including the target. This is defined as:

\newcommand{\glslongextraShortTargetFmt}[1]{% 
\glsentryitem{#1}\glstarget{#1}{\{\glsxtrshort\{noindex,hyper=false\}{#1}\}}% 
\glsxtrpostnamehook{#1} }

Note that the post-name hook is included.

\glslongextraLongFmt{⟨entry-label⟩}

This governs the way that the long form should be displayed. This is defined as:

\newcommand{\glslongextraLongFmt}[1]{% 
{\glsxtrlong\{noindex,hyper=false\}{#1}}\glspostdescription }

Note that the post-description hook is included.

\glslongextraSubShortTargetFmt{⟨level⟩}{⟨entry-label⟩}

This governs the way that the short form for child entries should be displayed, including the target. This is defined as:

\newcommand{\glslongextraSubShortTargetFmt}[2]{% 
\glssubentryitem{#2}\glstarget{#2}{\{\glsxtrshort\{noindex,hyper=false\}{#2}\}}% 
\glsxtrpostnamehook{#2} }
\glslongextraSubLongFmt\{(level)\}{(entry-label)}

This governs the way that the long form for child entries should be displayed. This is defined as:
\newcommand{\glslongextraSubLongFmt}[2]{\glslongextraLongFmt{#2}}

\glslongextraShortNoNameSetDescWidth

This is used to compute the value of \glsdescwidth and assumes that none of the short forms are wider than \glslongextraShortHeader.

\textit{abbr-short-long}

A two column style. The short form is in the left column. The long form is in the right. The location list isn’t shown.

The table header is given by:
\glslongextraShortLongTabularHeader

which shows the column headers with horizontal rules.

The table footer is given by:
\glslongextraShortLongTabularFooter

which just does a horizontal rule. With \texttt{longtable}, the table header and footer are set with:
\glslongextraShortLongHeader

which uses the above header and footer commands.

\textit{abbr-long-short}

A two column style. The short form is in the right column. The long form is in the left. The location list isn’t shown.
The table header is given by:

\glslongextraLongShortTabularHeader

which shows the column headers with horizontal rules.

The table footer is given by:

\glslongextraLongShortTabularFooter

which just does a horizontal rule. With longtable, the table header and footer are set with:

\glslongextraLongShortHeader

which uses the above header and footer commands.

### 8.7.3. glossary-topic package

The glossary-topic package provides glossary styles designed for hierarchical glossaries where the top-level entries are topic titles. This package automatically loads the multicol package. If the glossary-tree package is also loaded then commands like \glssetwidest can be used on these styles in much the same way as for the alttree style. If a widest value isn’t set then these styles behave more like the tree style.

This package provides styles designed for glossaries that are lists of topics. That is, the top-level entries are considered topic titles (which may or may not have an associated symbol or description) and the sub-entries are items within that topic. By default the location list isn’t shown for the top-level entries but is shown after the description for sub-entries (unless suppressed with nonumberlist or save-locations=false).

The following styles are provided:

- **topic**
  
  This style is similar to the tree style but the indentation doesn’t start until the second sub-item level. The top-level entries have the name displayed in a larger font with the description following in a new paragraph (see Example 147 on the next page). This style doesn’t support the nogroupskip setting.
Example 147: The topic style

Glossary

Pictograph
Picture or symbol representing a word or phrase.
- copy.
- cut.
- edit.
- paste.

Symbols
Mathematical constants or functions.
- \textbf{constant} \hspace{1em} a fixed quantity or numerical value.
  - \(\pi\) \hspace{1em} ratio of the length of the circumference of a circle to its diameter.
  - \(\sqrt{2}\) \hspace{1em} Pythagoras’ constant.
- \textbf{function} \hspace{1em} a rule that assigns a value to every element of the domain.
  - \(\cos(x)\) \hspace{1em} cosine.
  - \(\ln(x)\) \hspace{1em} natural logarithm.
- \textbf{parameter} \hspace{1em} a constant or variable that distinguishes a specific form.
  - \(x\) \hspace{1em} the abscissa value.
  - \(y\) \hspace{1em} the ordinate value.
  - \(z\) \hspace{1em} the applicate value.

This style is like the topic style but the sub-entries are placed inside a \texttt{multicols} environment (unlike styles such as \texttt{mcoltree} where the entire glossary content is within a single \texttt{multicols} environment). The environment name is supplied in the value of the command:

\begin{verbatim}
\gls[topicColsEnv]{multicols}
\end{verbatim}

This defaults to \texttt{multicols}. You can change this to the starred form. For example:

\begin{verbatim}
\renewcommand{\gls[topicColsEnv]}{multicols*}
\end{verbatim}

The number of columns is given by the command:

\begin{verbatim}
\gls[topicCols]
\end{verbatim}
This expands to 2, by default. This style is demonstrated in Example 148.

\begin{glossary}
\section*{Pictograph}
Picture or symbol representing a word or phrase.

\begin{itemize}
\item copy.
\item cut.
\item edit.
\item paste.
\end{itemize}

\section*{Symbols}
Mathematical constants or functions.

\begin{itemize}
\item constant \enspace a fixed quantity or numerical value.
  \begin{itemize}
  \item $\pi$ \enspace ratio of the length of the circumference of a circle to its diameter.
  \item $\sqrt{2}$ \enspace Pythagoras’ constant.
  \end{itemize}
\item function \enspace a rule that assigns a value to every element of the domain.
\end{itemize}

\parskip1em
\begin{itemize}
\item cos($x$) \enspace cosine.
\item ln($x$) \enspace natural logarithm.
\item parameter \enspace a constant or variable that distinguishes a specific form.
  \begin{itemize}
  \item $x$ \enspace the abscissa value.
  \item $y$ \enspace the ordinate value.
  \item $z$ \enspace the applicate value.
  \end{itemize}
\end{itemize}

Both styles can have a widest name set like the alttree style, using the commands provided by glossary–tree and glossaries–extra–stylemods or with the set–widest resource option. If a widest name is set, then the sub-entry names will be placed in a box of the given width otherwise they won’t be placed in a box. In Example 149 on the following page, the widest names have been set for level 1 and level 2 using:

\begin{itemize}
\item \glsetwidest[1]{parameter}
\item \glsetwidest[2]{\glsentryname{cosine}}
\end{itemize}

Note that this doesn’t change the indentation at the start of the level 2 items to match the width of the level 1 widest name.

Both of the above styles use the following commands.

% \glstopicParIndent

463
8. Defining and Displaying Glossaries

Example 149: The \texttt{topicxcols} style with the widest name set

\begin{center}
\textbf{Glossary}
\end{center}

\textbf{Pictograph}

Picture or symbol representing a word or phrase.

\begin{itemize}
  \item \texttt{\copy} copy.
  \item \texttt{\cut} cut.
  \item \texttt{\edit} edit.
  \item \texttt{\paste} paste.
\end{itemize}

\textbf{Symbols}

Mathematical constants or functions.

\begin{itemize}
  \item \texttt{\constant} a fixed quantity or numerical value.
  \item $\pi$ ratio of the length of the circumference of a circle to its diameter.
  \item $\sqrt{2}$ Pythagoras' constant.
  \item \texttt{\function} a rule that assigns a value to every element of the domain.
  \item $\cos(\texttt{x})$ cosine.
  \item $\ln(\texttt{x})$ natural logarithm.
  \item \texttt{\parameter} a constant or variable that distinguishes a specific form.
  \item $x$ the abscissa value.
  \item $y$ the ordinate value.
  \item $z$ the applicate value.
\end{itemize}

This command is a length that’s used for the paragraph indentation in any multi-paragraph description for top-level entries, but not for the first paragraph (at the start of the description) which isn’t indented.

\texttt{\glstopicSubIndent}

This command is a length that’s used to calculate the hanging indentation for sub-entries. The level 1 sub-entries don’t indent the name. Level $n$ sub-entries have the name indented by $(n-1) \times \texttt{\glstopicSubIndent}$. The hanging indent depends on whether or not a widest indent has been set for the level.

There is also a length for additional indentation used in the second paragraph onwards for child entries with multi-paragraph descriptions:

\texttt{\glstopicSubItemParIndent}
This is initialised to \parindent when glossary-topic is loaded.

\glsstopicInit

This hook is used at the start of the glossary. It does nothing by default.

Although the styles don’t support letter groups by default, if you have many topics (top-level entries) and you feel that it would help the reader to divide them up into headed letter groups, you can redefine:

\glsstopicGroupHeading{⟨group-label⟩}

This does nothing by default. If you want to redefine it, you can fetch the title corresponding to the group label with \glsxtrgetgrouptitle. For example:

\renewcommand*{\glsstopicGroupHeading}[1]{\glsxtrgetgrouptitle{#1}{\thisgrptitle}\section*{\thisgrptitle}}

Remember that if you are using bib2gls, you will need the --group or -g switch to support groups (see §8.4.1).

Sub-groups are only available with bib2gls and the group-level option. If they are supported, sub-group headings are formatted according to:

\glsstopicSubGroupHeading{⟨prev group level⟩}{⟨group level⟩}{⟨parent entry⟩}{⟨group-label⟩}

This formats the sub-group heading. Note that unlike \glsstopicGroupHeading this command does actually format the sub-group heading by default. This means that if you use group-level=all, the top-level groups won’t be displayed, but the sub-groups will be.

Top-level entries are formatted according to:

\glsstopicItem{⟨entry-label⟩}{⟨location list⟩}

This formats the name and (if provided) the symbol. The description (if set) will follow in a
8. Defining and Displaying Glossaries

new paragraph. At the start of `\glistopicItem`, a vertical space is added with:

\glistopicPreSkip

which defaults to `\medskip`. There is then a hook:

\glistopicMarker{⟨entry-label⟩}

which does nothing by default, but may be redefined. For example, to add a line to the table of contents. The name and symbol are set in the form of a title using:

\glistopicTitle{⟨entry-label⟩}

This uses `\Glossentryname` which uses sentence case. If there’s a symbol, this is added in parentheses. Both name and symbol (if present) are encapsulated by:

\glistopicTitleFont{⟨text⟩}

This uses a bold, large font by default.

If the entry has the description key set (tested with `\ifglistopishasdesc`) then a paragraph break is inserted followed by:

\glistopicMidSkip

which defaults to `\smallskip`. This is followed by the description which is formatted according to:

\glistopicDesc{⟨entry-label⟩}

This just does `\Glossentrydesc{entry-label}` followed by the post-description hook. There is then a paragraph break followed by:

\glistopicPostSkip

regardless of whether or not the description was displayed. This defaults to `\smallskip`. 

466
This is then followed by:

```
\glstopicLoc{(entry-label)\{location list\}}
```

which may be used to display the location list, but does nothing by default.

The sub-entries first set up the paragraph and hanging indentations using:

```
\glstopicAssignSubIndent{\langle level\rangle}
```

This uses:

```
\glstopicAssignWidest{\langle level\rangle}
```

to determine if a widest name has been set for the given level.

The sub-entry has its information displayed using:

```
\glstopicSubItem{\langle level\rangle}{\langle entry-label\rangle}{\langle location list\rangle}
```

This encapsulates the name with:

```
\glstopicSubNameFont{\langle text\rangle}
```

By default this just uses \textbf. This is followed by:

```
\glstopicSubItemSep
```

which defaults to \quad. The name and separator are passed in the \langle text\rangle argument of:

```
\glstopicSubItemBox{\langle level\rangle}{\langle text\rangle}
```

If a widest name was set for the given level, this will put \langle text\rangle inside a box of that width otherwise it just does \langle text\rangle.

This is followed by the symbol in parentheses if set. Then, if the description is set, the
description and post-description hook are displayed followed by:

\glistopicSubPreLocSep

(This command isn’t used if the description isn’t set.)

Finally the location list is displayed using:

\glistopicSubLoc\{⟨entry-label⟩\}{⟨location list⟩}

which just does ⟨location⟩ by default.

8.7.4. glossary–table package

The glossary–table package is new to version 1.49 and is still experimental. It automatically loads glossary-longbooktabs.

It doesn’t provide any general purpose styles, but instead provides one highly customized style table designed to work with a supplied command:

\printunsrttable[⟨options⟩]

The aim of this command is to show the glossary as a longtable, where the caption is what would normally be the glossary header (title). What’s normally the table of contents title (tocTitle) will be added to the list of tables instead.

This internally uses \printunsrtglossary with the style set to table and is intended for use with bib2gls. It supports up to 1 hierarchical level, but you will need the save-child-count resource option if you want the level 1 sub-entries to show. Deeper level entries are omitted. If you want any additional filtering, you can redefine:

\glistableiffilter\{⟨entry-label⟩\}{⟨true⟩}{⟨false⟩}

This command should do ⟨true⟩ if the entry identified by ⟨entry-label⟩ should be skipped, otherwise it should do ⟨false⟩. For example, the following will filter entries that have the category set to general:

\renewcommand{\glistableiffilter}[1]\
\glsifcategory{#1}{general}

Note that if this command is redefined to do neither ⟨true⟩ nor ⟨false⟩ or does both, it will interfere with the width calculations if par isn’t set to the default par=false.
8. Defining and Displaying Glossaries

Styles like `long` create a `longtable` with one entry per row. The table style allows multiple entries per row. A “block” indicates a block of columns used to format one entry. One row of the table contains multiple blocks.

An extra field may be added with the `other` key. If this value is empty, then no extra field will be added. Block styles that show the description will put this other field in before the description, in the same column as the description. Block styles that don’t show the description, will put the other field after the name, in the same column as the name. Some block styles, such as `symbol-name` and `symbol-other` put the other field in its own column.

The optional argument may have the options that can typically be passed to \texttt{\textbackslash printunsortedglossary}, except that the `nogroupskip`, `nonumberlist` and `style` options won’t have an effect. If you want the location list, it can simply be obtained from the `location` field in the appropriate style hook.

Additionally, the following options may also be used.

```
blocks=(n)
```

The value is the number of blocks in the table. The total number of columns in the table will be this value multiplied by the number of columns per block, which is determined by the block style. For example, the `name-desc` has two columns, so if there are three blocks then there will be a total of six columns.

```
header=(boolean)
```

A boolean option to determine whether or not to show the header row. The header text is produced with one of the following commands:

```
\glstablenameheader
```

Expands to the header for the name column. Just uses `\entryname` by default.

```
\glstabledescheader
```

Expands to the header for the description column for block styles like `name-desc` and `name-symbol-desc`. Just uses `\descriptionname` by default.

```
\glstablesymbolheader
```

Expands to the header for the symbol column for block styles like `name-symbol` and `name-symbol-desc`. Just uses `\symbolname` by default.
### Defining and Displaying Glossaries

<table>
<thead>
<tr>
<th>Command</th>
<th>Default/Initial</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glstableotherheader</td>
<td></td>
<td>Expands to the header for the other column. The default definition applies <code>\ MFUsentence-case</code> to the other field label.</td>
</tr>
<tr>
<td>rules=⟨boolean⟩</td>
<td>default: true; initial: true</td>
<td>A boolean option to determine whether or not to show the horizontal rules (provided by <code>booktabs</code>). If used with <code>header=true</code>, there will be a rule above and below the header row. If used with <code>header=false</code>, there will only be one rule at the top of the table. In both cases, there will be a rule at the bottom of the table.</td>
</tr>
<tr>
<td>blocksep=⟨alignment spec⟩</td>
<td>initial:</td>
<td>The value is inserted into the alignment specifier list between blocks. For example, the default value of the pipe character will insert a vertical line. Set this value to empty to remove it.</td>
</tr>
<tr>
<td>par=⟨value⟩</td>
<td>initial: false</td>
<td>Indicates whether or not the columns should be paragraphs. The value may be one of: <code>false</code>, <code>justified</code> or <code>ragged</code>. The default <code>par=false</code> will just use one of the column specifiers <code>l</code>, <code>r</code> or <code>c</code>. The other values will use the <code>p</code> specifier, in which case the column widths will be calculated.</td>
</tr>
<tr>
<td>other=⟨field-label⟩</td>
<td>initial: empty</td>
<td>This should be set to the internal field label of the other field or to empty if no other field should be included.</td>
</tr>
<tr>
<td>init=⟨⟨code⟩⟩</td>
<td>initial: empty</td>
<td>The <code>⟨⟨code⟩⟩</code> will be added shortly before <code>\printunsrtglossary</code> is called and any local changes will be scoped.</td>
</tr>
<tr>
<td>block-style=⟨value⟩</td>
<td>initial: name-desc</td>
<td></td>
</tr>
</tbody>
</table>

470
The block style, which may be one of the following:

- \textit{block-style\,=\,name\,-\,desc}

Blocks have two columns with the name in the first column of the block and the description in the second. This is the default style.

- \textit{block-style\,=\,desc\,-\,name}

Blocks have two columns with the description in the first column of the block and the name in the second.

- \textit{block-style\,=\,name\,-\,symbol}

Blocks have two columns with the name in the first column of the block and the symbol in the second.

- \textit{block-style\,=\,symbol\,-\,name}

Blocks have two columns with the symbol in the first column of the block and the name in the second.

- \textit{block-style\,=\,name\,-\,symbol\,-\,desc}

Blocks have three columns with the name in the first column of the block, the symbol in the second, and the description in the third.

- \textit{block-style\,=\,name\,-\,desc\,-\,symbol}

Blocks have three columns with the name in the first column of the block, the description in the second, and the symbol in the third.

- \textit{block-style\,=\,name}

Blocks have one column with the name followed by the symbol and description in parenthe-
8. Defining and Displaying Glossaries

ses, if they have been set.

\begin{description}
\item[\textbf{block-style=\texttt{name-other}}]
This is like \texttt{name-desc} but puts the other field in the second column. The description and symbol aren’t shown. Note that you will need to set the \texttt{other} key to the required field, otherwise you will end up with an empty column.

\item[\textbf{block-style=\texttt{other-name}}]
This is like \texttt{desc-name} but puts the other field in the second column. The description and symbol aren’t shown. Note that you will need to set the \texttt{other} key to the required field, otherwise you will end up with an empty column.

\item[\textbf{block-style=\texttt{symbol-other}}]
This is like \texttt{name-other} but shows the symbol instead of the name.

\item[\textbf{block-style=\texttt{other-symbol}}]
This is like \texttt{other-name} but shows the symbol instead of the name.
\end{description}

The block styles alter the way the table style sets up the longtable environment and the way that the entries are formatted. The top level glossary style command \texttt{\glossentry} is defined to do the block according to the designated block style and then it will follow the block with the child entries, if the \texttt{childlist} field has been set.

The \texttt{\subglossentry} command is redefined to do nothing, but it won’t be used as the child entries are all filtered out. If you don’t use the \texttt{save-child-count} resource option, no child entries will be shown. There’s no recursive descent down the hierarchical levels.

This means that the child entries will be listed at the end of the last column in the block. This can make the column quite wide. The child names aren’t displayed by default but supports the \texttt{subentrycounter} option. The child entries are listed in a tabular environment, which means they are contained in the same row as their parent and can’t be broken across a page.

The remaining commands are undocumented as they are liable to change.
The following example uses the name block style, which only has one column per block. The name is followed by the description in parentheses (if set), which is then followed by the child list. I redefined \glsnamefont to make the name appear in bold, to highlight it. I’ve used the \par=ragged option, otherwise the table will be too wide to fit the page.

\usepackage
[record,stylemods=table,subentrycounter]
{glossaries-extra}

\GlsXtrLoadResources[
  % entries in example-glossaries-childnoname.bib:
  src=example-glossaries-childnoname,
  selection=all,
  save-child-count=]
\begin{document}
\printunsrttable
[
  block-style=name,\par=ragged,
  preamble={Some preamble text},
  postamble={Some postamble text},
  init={%
\let\glsnamefont\textbf
\def\glstablenameheader{Summary}%
  }%
]
\end{document}

This creates a table with two entries per row.
Note that each row is as deep as the entry with the most children. So where a row has one column with two children and another with seven, the row is deep enough to accommodate the seven child entries, which leaves a gap below the smaller list of two children.
9. Accessibility Support

The glossaries package comes with a supplementary package glossaries-accsupp that helps provide accessibility support. The glossaries-extra package provides additional support, but only if the glossaries-accsupp package has already been loaded when the relevant commands are defined. The best and simplest way to do this is through the accsupp package option.

See the “Accessibility Support” chapter in the glossaries user guide for further information about glossaries-accsupp.

9.1. Abbreviations

The accessibility fields relating to abbreviations are shortaccess, shortpluralaccess, longaccess and longpluralaccess. These provide the replacement text for the corresponding short, shortplural, long and longplural fields. The access field provides the replacement text for the name field.

Some of these accessibility fields are automatically assigned by \newabbreviation if they haven’t been set.

\glsxtrassignactualsetup

This command is used to locally redefine common formatting commands so that they can be stripped to obtain only the text. You can add additional commands with \appto. For example, the following eccentric example has some strange styling in the abbreviation:

\newabbreviation{foo}{f\textsuperscript{o}\textsubscript{o}}{furry old otters}  

If an accessibility field is being automatically assigned with text obtained from the short value, then the subscript and superscript commands will need to be stripped. These need to be locally redefined to just do their arguments:

\appto\glsxtrassignactualsetup{%    \letcs{\textsuperscript}{@firstofone}%    \letcs{\textsubscript}{@firstofone}%}
9. Accessibility Support

The attributes that specifically relate to accessibility in abbreviations are listed below. The “actual short value” means the value obtained from the `short` value after any markup commands have have locally redefined using \glsxtrassignactualsetup. The actual short value may then be modified by these attributes. Similarly, for the “actual long value”.

Finally, if `shortaccess` hasn’t already been set, it will be set to:

\glsdefaultshortaccess\{ ⟨actual long⟩\}\{ ⟨actual short⟩\}

(with \glsdefaultshortaccess expanded). This command is provided by glossaries-accsupp and is defined to do just \{ ⟨actual long⟩\}. It was redefined by glossaries-extra v1.42 to do \{ ⟨actual long⟩\} \{ ⟨actual short⟩\}, but has been reverted back to its original definition in v1.49.

accessinsertdots= ⟨boolean⟩

If the `shortaccess` key hasn’t been set then this attribute will be checked. If true, the actual short value will have dots inserted (as per `insertdots`). Note that if this attribute hasn’t been set but `insertdots` is true (and the `shortaccess` key hasn’t been set), then the actual short value will also have dots inserted.

accessaposplural= ⟨boolean⟩

If the `shortpluralaccess` key hasn’t been set then this attribute will be checked. If true, the actual short plural value will have the apostrophe suffix (similar to `aposplural` but using \glsxtrabbrvpluralsuffix instead of \abbrvpluralsuffix). Note that if this attribute hasn’t been set but `aposplural` is true (and the `shortpluralaccess` key hasn’t been set), then the actual short plural value will also have the apostrophe suffix.

accessnoshortplural= ⟨boolean⟩

If the `shortpluralaccess` key hasn’t been set and the `accessaposplural` attribute hasn’t been set, then this attribute will be checked. If true, the actual short plural value will be the same as the singular (as `noshortplural`). Note that if this attribute hasn’t been set but `noshortplural` is true (and the `shortpluralaccess` key hasn’t been set), then the actual short plural value will also be the singular form.

nameshortaccess= ⟨boolean⟩

If the `access` key hasn’t been set and this attribute is true, then the `access` field will be set
9. Accessibility Support

to the same as the shortaccess.

\text{textshortaccess}=\langle boolean\rangle

If the textaccess key hasn’t been set and this attribute is true, then the textaccess field will be set to the same as the shortaccess. Additionally, if the pluralaccess key hasn’t been set, then it will be set to the same as the shortpluralaccess value.

\text{firstshortaccess}=\langle boolean\rangle

If the firstaccess key hasn’t been set and this attribute is true, then the firstaccess field will be set to the same as the shortaccess. Additionally, if the firstpluralaccess key hasn’t been set, then it will be set to the same as the shortpluralaccess value.

9.2. Accessibility Wrappers

The glossary style commands such as \glossentryname incorporate accessibility support by using the \glsaccess⟨field⟩ commands instead of the corresponding \glsentry⟨field⟩ commands.

If the glossaries-accsupp package hasn’t been loaded or if the relevant accessibility field hasn’t been set, these commands simply do the corresponding \glsentry⟨field⟩ command.

\glsaccessname{⟨entry-label⟩}

This shows the name field encapsulated with \glsnameaccessdisplay or just \glsentryname{⟨entry-label⟩}.

\Glsaccessname{⟨entry-label⟩}

As above but sentence case.

\GLSaccessname{⟨entry-label⟩}

As above but all caps.

\glsaccesstext{⟨entry-label⟩}
This shows the text field encapsulated with \glsaccessdisplay or just \glsentrytext{(entry-label)}.

\Glsaccess\{entry-label\}

As above but sentence case.

\GLSaccess\{entry-label\}

As above but all caps.

\glsaccessplural\{entry-label\}

This shows the plural field encapsulated with \glspluralaccessdisplay or just \glsentryplural\{entry-label\}.

\Glsaccessplural\{entry-label\}

As above but sentence case.

\GLSaccessplural\{entry-label\}

As above but all caps.

\glsaccessfirst\{entry-label\}

This shows the first field encapsulated with \glsfirstaccessdisplay or just \glsentryfirst\{entry-label\}.

\Glsaccessfirst\{entry-label\}

As above but sentence case.

\GLSaccessfirst\{entry-label\}
As above but all caps.

\glsaccessfirstplural{(entry-label)}

This shows the \texttt{firstplural} field encapsulated with \texttt{\glsfirstpluralaccessdisplay} or just \texttt{\glsentryfirstplural{(entry-label)}}.

\GLSaccessfirstplural{(entry-label)}

As above but sentence case.

\GLSaccessfirstplural{(entry-label)}

As above but all caps.

\glsaccesssymbol{(entry-label)}

This shows the \texttt{symbol} field encapsulated with \texttt{\glssymbolaccessdisplay} or just \texttt{\glsentrysymbol{(entry-label)}}.

\GLSaccesssymbol{(entry-label)}

As above but sentence case.

\GLSaccesssymbol{(entry-label)}

As above but all caps.

\glsaccesssymbolplural{(entry-label)}

This shows the \texttt{symbolplural} field encapsulated with \texttt{\glssymbolpluralaccessdisplay} or just \texttt{\glsentrysymbolplural{(entry-label)}}.

\GLSaccesssymbolplural{(entry-label)}
9. Accessibility Support

As above but sentence case.

\GLSaccesssymbolplural{⟨entry-label⟩}

As above but all caps.

\glsaccessdesc{⟨entry-label⟩}

This shows the description field encapsulated with \glsdescriptionaccessdisplay or just \glsentrydesc{⟨entry-label⟩}.

\Glsaccessdesc{⟨entry-label⟩}

As above but sentence case.

\GLSaccessdesc{⟨entry-label⟩}

As above but all caps.

\glsaccessdescplural{⟨entry-label⟩}

This shows the descriptionplural field encapsulated with \glsdescriptionpluralaccessdisplay or just \glsentrydescplural{⟨entry-label⟩}.

\Glsaccessdescplural{⟨entry-label⟩}

As above but sentence case.

\GLSaccessdescplural{⟨entry-label⟩}

As above but all caps.

\glsaccessshort{⟨entry-label⟩}
9. Accessibility Support

This shows the short field encapsulated with \glsshortaccessdisplay or just \glsentryshort\{\textit{entry-label}\}.

\Glsaccessshort\{\textit{entry-label}\}

As above but sentence case.

\GLSaccessshort\{\textit{entry-label}\}

As above but all caps.

\glsaccessshortpl\{\textit{entry-label}\}

This shows the shortplural field encapsulated with \glsshortpluralaccessdisplay or just \glsentryshortpl\{\textit{entry-label}\}.

\Glsaccessshortpl\{\textit{entry-label}\}

As above but sentence case.

\GLSaccessshortpl\{\textit{entry-label}\}

As above but all caps.

\glsaccesslong\{\textit{entry-label}\}

This shows the long field encapsulated with \glslongaccessdisplay or just \glsentrylong\{\textit{entry-label}\}.

\Glsaccesslong\{\textit{entry-label}\}

As above but sentence case.

\GLSaccesslong\{\textit{entry-label}\}
9. Accessibility Support

As above but all caps.

\glsaccesslongpl{⟨entry-label⟩}

This shows the \longplural\ field encapsulated with \glslongpluralaccessdisplay or just \glsentrylongpl{(entry-label)}.

\GLSaccesslongpl{⟨entry-label⟩}

As above but sentence case.

\GLSaccesslongpl{⟨entry-label⟩}

As above but all caps.

\glsaccessuseri{(entry-label)}

This shows the \useri\ field encapsulated with \glsuseriaccessdisplay or just \glsentryuseri{(entry-label)}.

\GLSaccessuseri{(entry-label)}

As above but sentence case.

\GLSaccessuseri{(entry-label)}

As above but all caps.

\glsaccessuserii{(entry-label)}

This shows the \userii\ field encapsulated with \glsuseriiaccessdisplay or just \glsentryuserii{(entry-label)}.

\GLSaccessuserii{(entry-label)}
9. Accessibility Support

As above but sentence case.

\GLSaccessuserii\{⟨entry-label⟩\}

As above but all caps.

\glsaccessuserii\{⟨entry-label⟩\}

This shows the user3 field encapsulated with \glsuseriiaccessdisplay or just \gls-\entryuserii\{⟨entry-label⟩\}.

\GLSaccessuserii\{⟨entry-label⟩\}

As above but sentence case.

\glsaccessuserii\{⟨entry-label⟩\}

As above but all caps.

\glsaccessuseriv\{⟨entry-label⟩\}

This shows the user4 field encapsulated with \glsuserivaccessdisplay or just \gls-\entryuseriv\{⟨entry-label⟩\}.

\GLSaccessuseriv\{⟨entry-label⟩\}

As above but sentence case.

\glsaccessuseriv\{⟨entry-label⟩\}

As above but all caps.

\glsaccessuserv\{⟨entry-label⟩\}
9. Accessibility Support

This shows the user5 field encapsulated with \glsuservaccessdisplay or just \glsentryuserv\{\langle entry-label\rangle\}.

\Glsaccessuserv\{\langle entry-label\rangle\}

As above but sentence case.

\GLSaccessuserv\{\langle entry-label\rangle\}

As above but all caps.

\glsaccessuservi\{\langle entry-label\rangle\}

This shows the user6 field encapsulated with \glsuserviaccessdisplay or just \glsentryuservi\{\langle entry-label\rangle\}.

\Glsaccessuservi\{\langle entry-label\rangle\}

As above but sentence case.

\GLSaccessuservi\{\langle entry-label\rangle\}

As above but all caps.

9.3. Inner Formatting Wrappers

These \glsaccessfmt\{field\} commands are similar to the corresponding \glsaccess\{field\} commands described above, but they format the field value using \glsfmtfield, \Glsfmtfield or \GLSfmtfield with the supplied \langle cs\rangle encapsulating command.

The default entry display style \glsgenentryfmt, and the predefined abbreviation styles all incorporate accessibility support by using these commands in order to support the inner formatting.

\glsaccessfmtname\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
This shows the `name` field formatted with `⟨cs⟩` and, if accessibility support provided, encapsulated with `\glsnameaccessdisplay`.

\Glsaccessfmtname{(insert)}{(cs)}{(entry-label)}

As above but sentence case.

\GLSaccessfmtname{(insert)}{(cs)}{(entry-label)}

As above but all caps.

\glsaccessfmttext{(insert)}{(cs)}{(entry-label)}

This shows the `text` field formatted with `⟨cs⟩` and, if accessibility support provided, encapsulated with `\glstextaccessdisplay`.

\Glsaccessfmttext{(insert)}{(cs)}{(entry-label)}

As above but sentence case.

\GLSaccessfmttext{(insert)}{(cs)}{(entry-label)}

As above but all caps.

\glsaccessfmtplural{(insert)}{(cs)}{(entry-label)}

This shows the `plural` field formatted with `⟨cs⟩` and, if accessibility support provided, encapsulated with `\glspluralaccessdisplay`.

\Glsaccessfmtplural{(insert)}{(cs)}{(entry-label)}

As above but sentence case.

\GLSaccessfmtplural{(insert)}{(cs)}{(entry-label)}
9. Accessibility Support

As above but all caps.

\glsaccessfmtfirst{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the first field formatted with ⟨cs⟩ and, if accessibility support provided, encapsulated with \glsfirstaccessdisplay.

\GlSaccessfmtfirst{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtfirst{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtfirstplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the firstplural field formatted with ⟨cs⟩ and, if accessibility support provided, encapsulated with \glsfirstpluralaccessdisplay.

\GlSaccessfmtfirstplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtfirstplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtsmbol{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the symbol field formatted with ⟨cs⟩ and, if accessibility support provided, encapsulated with \glssymbolaccessdisplay.

\GlSaccessfmtsmbol{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}
As above but sentence case.

\GLSaccessfmtsymbol\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

As above but all caps.

\glssaccessfmtsymbolplural\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

This shows the `symbolplural` field formatted with `\textit{cs}` and, if accessibility support provided, encapsulated with `\glssymbolpluralaccessdisplay`.

\GLSaccessfmtsymbolplural\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

As above but sentence case.

\GLSaccessfmtsymbolplural\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

As above but all caps.

\glsaccessfmtdesc\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

This shows the `description` field formatted with `\textit{cs}` and, if accessibility support provided, encapsulated with `\glsdescriptionaccessdisplay`.

\GLSaccessfmtdesc\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

As above but sentence case.

\GLSaccessfmtdesc\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

As above but all caps.

\glsaccessfmtdescplural\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}

This shows the `descriptionplural` field formatted with `\textit{cs}` and, if accessibility support
provided, encapsulated with \glsdescriptionpluralaccessdisplay.

\Glsaccessfmtdescplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtdescplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtshort{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the short field formatted with ⟨cs⟩ and, if accessibility support provided, encapsulated with \glsshortaccessdisplay.

\Glsaccessfmtshort{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtshort{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtshortpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the shortplural field formatted with ⟨cs⟩ and, if accessibility support provided, encapsulated with \glsshortpluralaccessdisplay.

\Glsaccessfmtshortpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtshortpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}
9. Accessibility Support

As above but all caps.

\glsaccessfmtlong{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the \texttt{long} field formatted with \texttt{⟨cs⟩} and, if accessibility support provided, encapsulated with \texttt{\glslongaccessdisplay}.

\GLSaccessfmtlong{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtlongpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtlongpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the \texttt{longplural} field formatted with \texttt{⟨cs⟩} and, if accessibility support provided, encapsulated with \texttt{\glslongpluralaccessdisplay}.

\GLSaccessfmtlongpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtuseri{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtuseri{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the \texttt{user1} field formatted with \texttt{⟨cs⟩} and, if accessibility support provided, encapsulated with \texttt{\glsuseriaccessdisplay}.

\GLSaccessfmtuseri{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}
9. Accessibility Support

As above but sentence case.

\GLSaccessfmtuseri{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtuserii{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the user2 field formatted with ⟨cs⟩ and, if accessibility support provided, encapsulated with \glsuseriiaccessdisplay.

\Glsaccessfmtuserii{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtuseri{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtuseriii{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

This shows the user3 field formatted with ⟨cs⟩ and, if accessibility support provided, encapsulated with \glsuseriiiaccessdisplay.

\Glsaccessfmtuseriii{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but sentence case.

\GLSaccessfmtuseriii{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}

As above but all caps.

\glsaccessfmtuseriv{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}
9. Accessibility Support

This shows the user4 field formatted with \langle cs⟩ and, if accessibility support provided, encapsulated with \glsuserivaccessdisplay.

\Glsaccessfmtuseriv\{insert\}\{cs\}\{entry-label\}

As above but sentence case.

\GLSaccessfmtuseriv\{insert\}\{cs\}\{entry-label\}

As above but all caps.

\glsaccessfmtuser\{insert\}\{cs\}\{entry-label\}

This shows the user5 field formatted with \langle cs⟩ and, if accessibility support provided, encapsulated with \glsuservaccessdisplay.

\Glsaccessfmtuserv\{insert\}\{cs\}\{entry-label\}

As above but sentence case.

\GLSaccessfmtuserv\{insert\}\{cs\}\{entry-label\}

As above but all caps.

\glsaccessfmtuservi\{insert\}\{cs\}\{entry-label\}

This shows the user6 field formatted with \langle cs⟩ and, if accessibility support provided, encapsulated with \glsuserviaccessdisplay.

\Glsaccessfmtuservi\{insert\}\{cs\}\{entry-label\}

As above but sentence case.

\GLSaccessfmtuservi\{insert\}\{cs\}\{entry-label\}

As above but all caps.

491
10. Categories

For multi-entry categories, see §7.8.

Each entry defined by \newglossaryentry (or commands that internally use it such as \newabbreviation) is assigned a category through the category key. You may add any category that you like, but since the category is a label used in the creation of some control sequences, avoid problematic characters within the category label. (So take care if you have babel shorthands on that make some characters active.)

The use of categories can give you more control over the way entries are displayed in the text or glossary. Note that an entry’s category is independent of the glossary type. Be careful not to confuse category with type.

\glscategory{(entry-label)}

Expands to the category of the given entry or does nothing if the entry doesn’t exist (analogous to \glsentryname).

\glsifcategory{(entry-label)}{(category)}{(true)}{(false)}

Tests if the entry given by (entry-label) has the category set to (category).

An entry may have its category field changed using commands such as \GlsXtrSetField (see §3.5). In addition, the following commands are provided to batch set the category for a collection of entries.

\glsxtrsetcategory{(entry-labels)}{(category-label)}

Globally sets the category field to the fully expanded (category-label) for each entry listed in (entry-labels).

\glsxtrsetcategoryforall{(glossary-labels)}{(category-label)}
10. Categories

Globally sets the category field to the fully expanded \langle category-label \rangle for each entry belonging to the glossaries listed in \langle glossary-labels \rangle.

There are also some iterative commands available:

\glsforeachincategory[\langle glossary-types \rangle]{\langle category \rangle}{\langle glossary-CS \rangle}{\langle label-CS \rangle}{\langle body \rangle}

This iterates through all entries in the glossaries identified by the comma-separated list \langle glossary-labels \rangle that have the category given by \langle category-label \rangle and performs \langle body \rangle for each match. Within \langle body \rangle, you can use \langle glossary-CS \rangle and \langle label-CS \rangle (which must be control sequences) to access the current glossary and entry label. If \langle glossary-labels \rangle is omitted, all glossaries are assumed.

\glsforeachwithattribute[\langle glossary-types \rangle]{\langle attribute-label \rangle}{\langle attribute-value \rangle}{\langle glossary-CS \rangle}{\langle label-CS \rangle}{\langle body \rangle}

This iterates over all entries in the given list of glossaries that have a category with the given \langle attribute-label \rangle set to \langle attribute-value \rangle and performs \langle body \rangle at each iteration. If \langle glossary-types \rangle is omitted, the list of all non-ignored glossaries is assumed. The remaining arguments are as for \glsforeachincategory.

10.1. Known Categories

These are the category labels that are set or referenced by glossaries-extra.

**general**

The default category assumed by \texttt{\newglossaryentry}.

**abbreviation**

The default category assumed by \texttt{\newabbreviation}.

**acronym**

The default category assumed by \texttt{\newacronym}.

**index**

The default category assumed by \texttt{\newindex}.
The default category assumed by \texttt{\newterm}.

\begin{itemize}
  \item \texttt{symbol}
  \end{itemize}

The default category assumed by \texttt{\glsxtrnewsymbol}.

\begin{itemize}
  \item \texttt{number}
  \end{itemize}

The default category assumed by \texttt{\glsxtrnewnumber}.

\section{10. Categories}

Each category may have a set of attributes, where each attribute has an associated value for its given category. An entry's attribute set corresponds to the attributes associated with the entry's category.

As with the category, the attribute name is also a label. You can provide your own custom attributes, which you can set and access with the commands described in §10.2.2.

\subsection{10.2. Known Attributes}

This section lists attributes that glossaries-extra sets or accesses. If an attribute hasn't been set, a default is assumed. For boolean attributes, the test may simply be to determine if the attribute has been set to \texttt{true}, in which case any other value or a missing value will be interpreted as false. Conversely, the test may be to determine if the attribute has been set to \texttt{false}, in which case any other value or a missing value will be interpreted as true.

See §7.8 for attributes relating to multi-entry categories.

\subsection{10.2.1. Abbreviation Attributes}

See §9.1 for abbreviation accessibility attributes.

\begin{itemize}
  \item \texttt{regular=\langle boolean\rangle}
  \end{itemize}

This attribute indicates whether or not an entry should be considered a regular entry. This enables \texttt{\glsentryfmt} to determine whether to use \texttt{\glsgenentryfmt} or \texttt{\glsxtrgenabbrvfmt}.
The `general` and `acronym` categories have the `regular` attribute automatically set to `true`. Some abbreviation styles change this value.

`discardperiod=⟨boolean⟩`

If set to “true”, the post-link hook will discard a full stop that follows `non-plural` commands like \gls or \glstext (see §5.5.4).

This attribute doesn’t apply to the accessibility fields. See §9.1 for attributes related to accessibility support for abbreviations.

Note that this can cause a problem if you access a field that doesn’t end with a full stop. For example:

\newabbreviation
  [user1={German Speaking \TeX\ User Group}]
  {dante} {DANTE e.V.} {Deutschsprachige Anwendervereinigung \TeX\ e.V.}

Here the `short` and `long` fields end with a full stop, but the `user1` field doesn’t. The simplest solution in this situation is to put the sentence terminator in the final optional argument. For example:

\glsuseri{dante}[.]

This will bring the punctuation character inside the link text and it won’t be discarded.

`pluraldiscardperiod=⟨boolean⟩`

If this attribute is set to “true” and the `discardperiod` attribute is set to “true”, this will behave as above for the plural commands like \glspl or \glsplural.

`retainfirstuseperiod=⟨boolean⟩`

If this attribute is set to “true” then the discard is determined by \glsxtrdiscardperiod-retainfirstuse, regardless of the `discardperiod` or `pluraldiscardperiod` attributes. This is useful for ⟨short⟩ ⟨⟨long⟩⟩ abbreviation styles where only the short form has a trailing full stop.
This attribute doesn’t apply to the accessibility fields. See §9.1 for attributes related to accessibility support for abbreviations.

\textbf{markwords=⟨boolean⟩}

If this attribute is set to “true” any entry defined using \newabbreviation will automatically have spaces in the long form replaced with:

\textbf{\glsxtrwordsep}

and each word is encapsulated with:

\textbf{\glsxtrword{⟨word⟩}}

For example:

\Glssetcategoryattribute{abbreviation}{markwords}{true}
\newabbreviation{ip}{IP}{Internet Protocol}

is essentially the same as

\newabbreviation{ip}{IP}{\glsxtrword{Internet}\glsxtrwordsep\glsxtrword{Protocol}}

The “hyphen” styles, such as \texttt{long-hyphen-short-hyphen}, take advantage of this markup. If the inserted material (provided in the final argument of \gls-like commands) starts with a hyphen then \glsxtrwordsep is locally redefined to a hyphen. (The default value is a space). Note that this only applies to commands like \gls and not like \glsxtrlong. You can provide your own localised switch, if required. For example:

\newcommand{\hyplong}[2][]{%
{\def\glsxtrwordsep{-}\glsxtrlong[#1]{#2}}}

This setting will also adjust the long plural. This attribute is only applicable to entries defined using \newabbreviation (or \newacronym if it’s using \newabbreviation.)
10. Categories

This setting may result in the \glsxtrword and \glsxtrwordsep markup ending up in the sort field, depending on the style in use.

markshortwords=(boolean)

This is similar to markwords but applies to the short form. (Only useful for abbreviations that contain spaces.) This attribute is only applicable to entries defined using \newabbreviation (or \newacronym if it’s using \newabbreviation.)

This setting will only adjust the short plural if the shortplural key isn’t used. This setting will take precedence over insertdots.

This setting may result in the \glsxtrword and \glsxtrwordsep markup ending up in the sort field, depending on the style in use.

insertdots=(boolean)

If this attribute is set to “true” any entry defined using \newabbreviation will automatically have full stops inserted after each letter. The entry will be defined with those dots present as though they had been present in the ⟨short⟩ argument of \newabbreviation (rather than inserting them every time the entry is used). The short plural form defaults to the new dotted version of the original ⟨short⟩ form with the plural suffix appended. This setting is incompatible with markshortwords. This attribute is only applicable to entries defined using \newabbreviation (or \newacronym if it’s using \newabbreviation.)

If you explicitly override the short plural using the shortplural key, you must explicitly insert the dots yourself (since there’s no way for the code to determine if the plural has a suffix that shouldn’t be followed by a dot).

This attribute is best used with the discardperiod attribute set to “true”.

aposplural=(boolean)

If this attribute is set to “true”, \newabbreviation will insert an apostrophe (’) before the plural suffix for the short plural form (unless explicitly overridden with the shortplural key). The long plural form is unaffected by this setting. This setting overrides noshortplural. This attribute is only applicable to entries defined using \newabbreviation (or
10. Categories

\newacronym if it’s using \newabbreviation.) Check with your supervisor, publisher or editor if you want to use this attribute as this usage is controversial.

\textbf{no\textit{short}plural}=(boolean)

If this attribute is set to “true”, \newabbreviation won’t append the plural suffix for the short plural form. This means the \textit{short} and \textit{shortplural} values will be the same unless explicitly overridden. \textit{This setting is incompatible with \textit{aposplural}. This attribute is only applicable to entries defined using \newabbreviation (or \newacronym if it’s using \newabbreviation.)}

\textbf{tagging}=(boolean)

If this attribute is set to “true”, the tagging command defined by \Gls\textit{XtrEnableInitial-Tagging} will be activated to use \texttt{glsxtrtagfont} in the glossary (see §4.4)

\textbf{10.2.1.2. Attributes that Alter \textit{glslink} Options}

\textbf{no\textit{hyper}first}=(boolean)

When used with the \texttt{\textit{gls}-like commands}, if this attribute is set to \texttt{true}, this will automatically suppress the hyperlink on first use.

This settings can be overridden by explicitly setting the \texttt{hyper} key on or off in the optional argument of the \texttt{\textit{gls}-like command}.

As from version 1.07, \textit{glsfirst}, \texttt{\textit{Glsfirst}}, \texttt{\textit{GLSfirst}} and their plural versions (which should ideally behave in a similar way to the first use of \textit{gls} or \textit{glspl}) now honour this attribute (but not the package-wide \texttt{hyperfirst=false} option, which matches the behaviour of glossaries). If you want commands like \texttt{\textit{glsfirst}} to ignore the \texttt{no\textit{hyper}first} attribute then just redefine \texttt{\textit{glsxtrcheckno\textit{hyper}first}} to do nothing.

\textbf{no\textit{hyper}next}=(boolean)

If set to \texttt{true}, this will automatically set \texttt{hyper=false} on subsequent use when using the
\texttt{\textbackslash nohyper=}\langle boolean\rangle

If set to \texttt{true}, this will automatically set \texttt{\hyper}=false when using the \texttt{\textbackslash gls}-like or \texttt{\textbackslash glstext}-like commands.

\texttt{\textbackslash indexonlyfirst=}\langle boolean\rangle

This is similar to the \texttt{\textbackslash indexonlyfirst} package option but only for entries that have a category with this attribute set to “true”.

\texttt{\textbackslash wr gloss=}\langle value\rangle

When using the \texttt{\textbackslash gls}-like or \texttt{\textbackslash glstext}-like commands, this will automatically set \texttt{\textbackslash wr gloss} = \texttt{after} if this attribute is set to “after”.

\texttt{\textbackslash textformat=}\langle cs-name\rangle

The \texttt{\textbackslash gls}-like and \texttt{\textbackslash glstext}-like commands have the link text encapsulated in the argument of \texttt{\textbackslash glstextformat} by default (the outer formatting, see §5.5.1). If the \texttt{\textbackslash textformat} attribute is set, the control sequence given by the attribute value will be used instead. The attribute value should be the name (without the leading backslash) of a command that takes a single argument (the link text). Remember that the abbreviation styles may apply an additional font change.

\texttt{\textbackslash hyperoutside=}\langle boolean\rangle

This boolean attribute may be \texttt{false}, \texttt{true} or unset. If unset, \texttt{true} is assumed. This indicates the default setting of the \texttt{\hyperoutside} option, described in §5.1.

10.2.1.3. Glossary Attributes

\texttt{\textbackslash glossdesc=}\langle value\rangle

This attribute is checked by the \texttt{\glossentrydesc} to determine whether or not to apply any case change. The value may be one of:

\texttt{firstuc}
10. Categories

Applies sentence case. That is, the first letter of the description will be converted to uppercase (using \Glsentrydesc).

title

Applies title case. If you have at least glossaries v4.48, the title casing is indirectly performed by \glscapitalisewords, which defaults to \capitalisewords (provided by mfirstuc). You can either redefine \glscapitalisewords if you want the change to also affect \glsentrytitlecase or if you only want the change to apply to the attribute case-changing then redefine \glsxtrfieldtitlecasecs. For example:

\begin{quote}
\newcommand*{\glsxtrfieldtitlecasecs}[1]{\xcapitalisefmtwords*{#1}}
\end{quote}

(Note that the argument to \glsxtrfieldtitlecasecs will be a control sequence whose replacement text is the entry’s description, which is why \xcapitalisefmtwords is needed instead of \capitalisefmtwords.)

If an error occurs with this setting, try redefining \glsxtrfieldtitlecasecs as shown above.

Any other values of this attribute are ignored. Remember that there are design limitations for both the sentence case and the title case commands. See the mfirstuc user manual for further details.

If you are using bib2gls, you can use the description-case-change setting instead.

\begin{enumerate}
\item \texttt{glossdescfont=⟨cs-name⟩}
\end{enumerate}

If set, the value should be the name of a control sequence (without the leading backslash) that takes one argument. This control sequence will be applied by \glossentrydesc to the description text. For example:

\begin{quote}
\glssetcategoryattribute{general}{glossdescfont}{emph}
\end{quote}

\begin{enumerate}
\item \texttt{glossname=⟨value⟩}
\end{enumerate}
As \texttt{glossdesc} but applies to \texttt{glossentryname}. Additionally, if this attribute is set to “uc” the name is converted to all caps.

If you are using \texttt{bib2gls}, you can use the \texttt{name-case-change} setting instead.

\texttt{indexname=\langle value\rangle}

If set, the post-name hook will index the entry using \texttt{index}. See §12 for further details.

\texttt{glossnamefont=\langle cs-name\rangle}

As \texttt{glossdescfont} but applies to \texttt{glossentryname}. Note that this overrides \texttt{glsnamefont} which will only be used if this attribute hasn’t been set.

Remember that glossary styles may additionally apply a font change, such as the list styles which put the name in the optional argument of \texttt{item}.

\texttt{glosssymbolfont=\langle cs-name\rangle}

This is similar to \texttt{glossnamefont} and \texttt{glossdescfont} but is used by \texttt{glossentrysymbol}.

10.2.1.4. Other Attributes

\texttt{headuc=\langle boolean\rangle}

If this attribute is set to “true”, commands like \texttt{glsfmtshort} will use the upper case version in the page headers.

\texttt{entrycount=\langle trigger-value\rangle}

The value of this attribute (if set) must be an integer and is used in combination with \texttt{gls-enableentrycount} (see §6.1). Leave blank or undefined for categories that shouldn’t have this facility enabled. The value of this attribute is used by \texttt{glsxtrifcounttrigger} to determine how commands such as \texttt{cgls} should behave.
10. Categories

This attribute is set to true by $\texttt{GlsXtrEnableLinkCounting}$ (see §6.2).

\begin{verbatim}
linkcountmaster=⟨counter-name⟩
\end{verbatim}

This attribute is set by $\texttt{GlsXtrEnableLinkCounting}$ to the name of the counter that requires the link counter to be added to its reset list (see §6.2).

\begin{verbatim}
dualindex=⟨value⟩
\end{verbatim}

If this attribute is set, whenever a glossary entry has information written to the external glossary file through commands like \texttt{\gls} and \texttt{\glsadd}, a corresponding line will be written to the indexing file using \texttt{\index}. The value may be \texttt{true} to simply enable this feature or the value may be the encap to use with \texttt{\index}. See §12 for further details.

\begin{verbatim}
targeturl=⟨url⟩
\end{verbatim}

If set, the hyperlink generated by commands like \texttt{\gls} will be set to the URL provided by this attribute’s value. For example:

\begin{verbatim}
\texttt{\glssetcategoryattribute{general}{targeturl}{master-doc.pdf}}
\end{verbatim}

(See also the accompanying sample file \texttt{sample-external.tex}.) If the URL contains awkward characters (such as % or \texttt{~}) remember that the base glossaries package provides commands like \texttt{\glspercentchar} and \texttt{\glstildechar} that expand to literal characters.

\begin{verbatim}
targetname=⟨anchor⟩
\end{verbatim}

If you want to a named anchor within the target URL (notionally adding \#⟨name⟩ to the URL), then you also need to set \texttt{targetname} to the anchor ⟨name⟩. You may use \texttt{\glslabel} within ⟨name⟩ which is set by commands like \texttt{\gls} to the entry’s label.

All the predefined glossary styles start each entry listing with \texttt{\glstarget} which sets the anchor to \texttt{\glolinkprefix\glslabel}, so if you want entries to link to glossaries in the URL given by \texttt{targeturl}, you can just do:

\begin{verbatim}
\texttt{\glssetcategoryattribute{general}{targetname}{\glolinkprefix\glslabel}}
\end{verbatim}

(If the target document changed \texttt{\glolinkprefix} then you will need to adjust the above as
appropriate.)

\begin{verbatim}
<anchor>
\end{verbatim}

If the anchor is in the form \langle name1 \rangle . \langle name2 \rangle then use targetname for the \langle name2 \rangle part and targetcategory for the \langle name1 \rangle part.

For example:

\begin{verbatim}
\glssetcategoryattribute{general}{targeturl}{master-doc.pdf}
\glssetcategoryattribute{general}{targetcategory}{page}
\glssetcategoryattribute{general}{targetname}{7}
\end{verbatim}

will cause all link text for general entries to link to master-doc.pdf#page.7 (page 7 of that PDF).

If you want a mixture in your document of entries that link to an internal glossary and entries that link to an external URL then you can use \texttt{\newignoredglossary} for the external list. For example:

\begin{verbatim}
\newignoredglossary*{external}
\glssetcategoryattribute{external}{targeturl}{master-doc.pdf}
\glssetcategoryattribute{general}{targetname}{\glolinkprefix\glslabel}
\newglossaryentry{sample}{name={sample},description={local example}}
\newglossaryentry{sample2}{name={sample2},
    type={external},
    category={external},
    description={external example}}
\end{verbatim}

\begin{verbatim}
externallocation=(PDF filename)
\end{verbatim}

The value should be the file name of the target document when manually indexing an external location with \texttt{thevalue}. In general, it’s better to use \texttt{bib2gls v1.7+} which can handle multiple external sources and doesn’t require this attribute.
10.2.2. Accessing and Setting Attributes

Attributes can be set using the following commands:

\glssetcategoryattribute{⟨category⟩}{⟨attribute⟩}{⟨value⟩}

Locally sets the given attribute to ⟨value⟩ for the given category.

\glssetcategoriesattribute{⟨category list⟩}{⟨attribute⟩}{⟨value⟩}

Globally sets the given attribute to ⟨value⟩ for all the categories in the comma-separated list ⟨category list⟩.

\glssetcategoryattributes{⟨category⟩}{⟨attribute list⟩}{⟨value⟩}

Globally sets each attribute in the comma separated ⟨attribute list⟩ to ⟨value⟩ for the given ⟨category⟩.

\glssetcategoriesattributes{⟨category list⟩}{⟨attribute list⟩}{⟨value⟩}

Globally sets each attribute in the comma separated ⟨attribute list⟩ to ⟨value⟩ for each category in the comma-separated list ⟨category list⟩.

\glssetattribute{⟨entry-label⟩}{⟨attribute⟩}{⟨value⟩}

Locally sets the given attribute to ⟨value⟩ for the category associated with the entry identified by ⟨entry-label⟩. This command can’t be used to assign an attribute for a multi-entry category.

\glssetregularcategory{⟨category⟩}

A shortcut that sets the regular attribute to true for the given category using \glssetcategoryattribute.

An attribute can be locally unset using:

\glsunsetcategoryattribute{⟨category⟩}{⟨attribute⟩}
10. Categories

Attribute values can be obtained with the following commands:

\glsgetcategoryattribute{⟨category⟩}{⟨attribute⟩}

Expands to the value of the given attribute for the given category. Expands to nothing if the attribute hasn’t been set.

\glsgetattribute{⟨entry-label⟩}{⟨attribute⟩}

Expands to the value of the given attribute for the category associated with the entry identified by ⟨entry-label⟩. Expands to nothing if the attribute hasn’t been set. This command can’t be used to assign an attribute for a multi-entry category.

Attributes can be tested with the following commands.

\glshascategoryattribute{⟨category⟩}{⟨attribute⟩}{⟨true⟩}{⟨false⟩}

This uses etoolbox’s \ifcsvoid and does ⟨true⟩ if the attribute has been set and isn’t blank and isn’t \relax otherwise it does ⟨false⟩.

\glshasattribute{⟨entry-label⟩}{⟨attribute⟩}{⟨true⟩}{⟨false⟩}

As \glshascategoryattribute but the category is obtained from the given entry. This command can’t be used to test an attribute associated with a multi-entry category.

\glsifcategoryattribute{⟨category⟩}{⟨attribute⟩}{⟨value⟩}{⟨true⟩}{⟨false⟩}

This tests if the given attribute for the given category is set and equal to ⟨value⟩. If true, ⟨true⟩ is done. If the attribute isn’t set or is set but isn’t equal to ⟨value⟩, ⟨false⟩ is done.

For example:

\glsifcategoryattribute{general}{nohyper}{true}{NO HYPER}{HYPER}

This does “NO HYPER” if the general category has the nohyper attribute set to true otherwise if does “HYPER”.

\glsifattribute{⟨entry-label⟩}{⟨attribute⟩}{⟨value⟩}{⟨true⟩}{⟨false⟩}
As `\glsifcategoryattribute` but the category is obtained from the given entry. This command can’t be used to test an attribute associated with a multi-entry category.

\begin{verbatim}
\glsifregularcategory\{\langle category \rangle\}\{(true)\}\{(false)\}
\end{verbatim}

A shortcut that tests if the given category has the `regular` attribute set to `true`.

\begin{verbatim}
\glsifnotregularcategory\{\langle category \rangle\}\{(true)\}\{(false)\}
\end{verbatim}

A shortcut that tests if the given category has the `regular` attribute set to `false`.

\textcolor{red}{\textbf{⚠️}} If the `regular` attribute hasn’t been set, both `\glsifregularcategory` and `\glsifnotregularcategory` will do `(false)`. The choice of command needs to be determined by what outcome should occur if the attribute hasn’t been set.

\begin{verbatim}
\glsifregular\{\langle entry-label \rangle\}\{(true)\}\{(false)\}
\end{verbatim}

As `\glsifregularcategory` but the category is obtained from the given entry. This command can’t be used to test an attribute associated with a multi-entry category.

\begin{verbatim}
\glsifnotregular\{\langle entry-label \rangle\}\{(true)\}\{(false)\}
\end{verbatim}

As `\glsifnotregularcategory` but the category is obtained from the given entry. This command can’t be used to test an attribute associated with a multi-entry category.

\begin{verbatim}
\glsifcategoryattributetrue\{\langle category \rangle\}\{\langle attribute \rangle\}\{(true)\}\{(false)\}
\end{verbatim}

Expands to `(true)` if the attribute is true and `(false)` otherwise. Expands to `(false)` if there’s no such attribute for the given category.

\begin{verbatim}
\glsifattributetrue\{\langle entry-label \rangle\}\{\langle attribute \rangle\}\{(true)\}\{(false)\}
\end{verbatim}

As `\glsifcategoryattributetrue` but the category is obtained from the given entry. Expands to `(false)` if the entry isn’t defined. This command can’t be used to test an attribute associated with a multi-entry category.
associated with a multi-entry category.

\[ \text{\texttt{\textbackslash glsif\texttt{categoryattributehasitem\{\texttt{category}\}\{\texttt{attribute}\}\{\texttt{item}\}\{\texttt{true}\}}} \{\texttt{false}\}}] 

Does \texttt{true} if the category has the attribute (whose value is a comma-separated list) contains the given item and \texttt{false} otherwise. Does \texttt{false} if there’s no such attribute for the given category. The item and list are expanded and passed to datatool’s \texttt{\textbackslash DTLifinlist} to perform the test.
11. bib2gls: Managing Reference Databases

The command line application bib2gls performs two functions in one:

- selects entries according to records found in the aux file (similar to Bib\TeX),
- hierarchically sorts entries and collates location lists (similar to makeindex or xindy).

Instead of storing all your entry definitions in a tex and loading them using \input or \loadglsentries, the entries can instead be stored in a bib file and bib2gls can selectively write the appropriate commands to a glstex file which is loaded using \GlsXtr-LoadResources.

This means that you can use a reference managing system to maintain the database and it reduces the \TeX overhead by only defining the entries that are actually required in the document. If you currently have a tex file that contains hundreds of definitions, but you only use a dozen or so in your document, then the build time is needlessly slowed by the unrequired definitions that occur when the file is input. (You can convert an existing tex file containing glossary definitions to a bib file using convertgls2bib, supplied with bib2gls.)

There are some new commands and options added to glossaries-extra to help assist the integration of bib2gls into the document build process.

This chapter just provides a general overview of bib2gls. The full details and some sample documents are provided in the bib2gls manual\footnote{mirrors.ctan.org/support/bib2gls/bib2gls.pdf}.

texdoc bib2gls

An example of the contents of bib file that stores glossary entries that can be extracted with bib2gls called, say, terms.bib:

```plaintext
@entry{bird,
name={bird},
description={feathered animal},
see={[see also]duck,goose}
}
```

\footnote{mirrors.ctan.org/support/bib2gls/bib2gls.pdf}
The following provides some abbreviations in a file called, say, abbrvs.bib:

```bib
text
@string{ssi={server-side includes}}
@string{html={hypertext markup language}}

@abbreviation{shtml,
  short="shtml",
  long= ssi # " enabled " # html,
  description={a combination of \gls{html} and \gls{ssi}}}

@abbreviation{html,
  short ="html",
  long = html,
  description={a markup language for creating web pages}}

@abbreviation{ssi,
  short="ssi",
  long = ssi,
  description={a simple interpreted server-side scripting language}}

@abbreviation{xml,
  short={xml}, long={extensible markup language}, description={a simple text-base format for representing structured information}}
```

The above defines bib strings (with @string) and uses string concatenation (with #), which is a \LaTeX{} feature. Another supported bib feature is @preamble, which may be used to provide command definitions.

Here are some symbols in a file called, say, symbols.bib:
To ensure that `bib2gls` can find out which entries have been used in the document, you need the `record` package option:

\usepackage[record]{glossaries-extra}

If you are using hyperref, you may prefer to use `record=nameref`. The `glstex` file created by `bib2gls` is loaded using:

\glsxtrresourcefile[⟨options⟩]{⟨basename⟩}

where `⟨basename⟩` is the basename (without the extension) of the `glstex` file. This command will redefine \glsindexingsetting to `bib2gls` (or `bib2gls-xindy` or `bib2gls-makeindex` if `record=hybrid`). There's a shortcut version:

\GlsXtrLoadResources[⟨options⟩]

This internally uses \glsxtrresourcefile and sets the `⟨basename⟩` to \jobname in the first instance and to \jobname-⟨n⟩ on subsequent instances (where ⟨n⟩ is incremented at the end of every \GlsXtrLoadResources). For example:
\usepackage[record]{glossaries-extra}
\GlsXtrLoadResources[src={terms,moreterms}]
\GlsXtrLoadResources[src={symbols,constants}]
\GlsXtrLoadResources[src={abbreviations}]

is equivalent to:

\usepackage[record]{glossaries-extra}
\glsxtrresourcefile[src={terms,moreterms}]{\jobname}
\glsxtrresourcefile[src={symbols,constants}]{\jobname-1}
\glsxtrresourcefile[src={abbreviations}]{\jobname-2}

If required, the value of \langle n \rangle is stored in the count register:

\glsxtrresourcecount

although there should be little need to use this.

Since \GlsXtrLoadResources is more convenient to use than \glsxtrresourcefile, all examples use \GlsXtrLoadResources.

The \glsxtrresourcefile command writes the following to the aux file:

\glsxtr@resource{\langle options \rangle}{\langle basename \rangle}

and will input \langle filename \rangle.glstex if it exists. (Version 1.08 assumed \langle filename \rangle.tex but that’s potentially dangerous if, for example, \langle filename \rangle happens to be the same as \jobname. The glstex extension was enforced by version 1.11.)

If you are using or developing a build system that needs to know which applications to run as part of the document build, you can search the aux for for instances of \glsxtr@resource. For example, using arara:

```latex
% arara: bib2gls if found(\"aux\", \"glsxtr@resource\")
```

Since the glstex file won’t exist on the first \LaTeX run, the record package option additionally switches on undefaction=warn. Any use of commands like \gls or \glstext will produce ?? in the document, since the entries are undefined at this point. Once bib2gls has created the glstex file the references should be resolved. This may cause a shift in the
locations if the actual text produced once the entry is defined is significantly larger than the placeholder ?? (as this can alter the page breaking).

Note that as from v1.12, \glsxtrresourcefile temporarily switches the category code of 0 to 11 (letter) while it reads the file to allow for any internal commands.

The package options record=only and record=nameref automatically load glossaries -extra-bib2gls, which provides additional commands that are useful with bib2gls. Since they cover sorting and location lists, they’re not relevant with the record=hybrid option.

These commands are provided by glossaries-extra for use with bib2gls.

The information provided with \GlsXtrLoadResources is written to the aux file using:

\protected@write@auxout{\glsxtrresourceinit}{⟨information⟩}

where ⟨information⟩ is the information to pass to bib2gls. The command in the second argument:

\glsxtrresourceinit

may be used to temporarily redefine commands before the information is written to the file. This does nothing by default, but may be redefined to allow the use of short commands for convenience. For example, with:

\renewcommand{\glsxtrresourceinit}{\let\u\glshex}

you can just use, for example, \u E6 instead of \string\uE6 in the custom rule. This redefinition of \u is scoped so its original definition is restored after the write operation.

\glsxtrMFUsave

If you have mfirstuc v2.08+, this command will be used on the first instance of \glsxtrresourcefile, and will add \MFUsave to the begin document hook and then disable itself. This is provided to help bib2gls v3.0+ pick up any of mfirstuc’s exclusions, blockers and mappings to assist with its sentence case function. The assumption is that all exclusions, blockers and mappings will be set up in the preamble. If there are any within the document environment that you want bib2gls to be aware of, redefine this command to do nothing before the first instance of \glsxtrresourcefile (or \GlsXtrLoadResources) and use \MFUsaveatend instead.
If you have multiple resource commands and you want a default set of options you can supply them in the definition of:

\\texttt{\textbackslash GlsXtrDefaultResourceOptions}

For example:

\\texttt{\renewcommand\{\textbackslash GlsXtrDefaultResourceOptions\}\{selection=all\}}

This should be done before the resource commands to which the options should apply.

\section*{11.1. Indexing (Recording)}

As with makeindex and xindy, the \texttt{\textbackslash gls}-like and \texttt{\textbackslash glstext}-like commands automatically index, but the underlying indexing mechanism is more like that used with \texttt{\textbackslash makenoidxglossaries}. Each indexing instance creates a record in the aux file, which bib2gls can then pick up when it parses the aux file. Each record has an associated format (the location encap) which can be set with the \texttt{format} key and an associated location counter (as with the other indexing methods).

The formatted location list is stored in the \texttt{location} field (unless \texttt{save-locations=false}). Additionally, the individual locations are stored in the \texttt{loclist} field as an etoolbox internal list (as with \texttt{\textbackslash makenoidxglossaries}). This may be used to pick out individual locations to avoid the complexity of parsing the formatted list.

See the bib2gls manual for information on how to separate the location list into groups associated with different counters.

\section*{11.2. Selection}

The default behaviour is for bib2gls to select all entries that have a record in the aux file, and any dependent entries (including parent and cross-references). The \texttt{glsignore} format (for example, \texttt{\textbackslash gls[format=glsignore]\{duck\}}) is recognised by bib2gls as a special ignored record. This means that it will match the selection criteria but the record won’t be added to the location list. This means that you won’t get spurious commas in the location list (as can happen with the other indexing methods), so you can do, for example,

\\texttt{\textbackslash GlsXtrSetDefaultNumberFormat\{glsignore\}}

at the start of the front matter and
at the start of the main matter to prevent any records in the front matter from occurring in the location lists.

Commands like \glsaddall and \glsaddallunused don’t work with bib2gls as the command has to iterate over each glossary’s internal lists of defined entry labels, which will be empty on the first run and on subsequent runs will only contain those entries that have been selected by bib2gls. Use selection=all to select all entries instead.

The selection option indicates which entries should be selected from the bib files (listed in src). For example, selection=all indicates to select all entries, regardless of whether or not the entries have been referenced in the document. This will lead to empty location lists for some (or all) entries. The default setting is selection=recorded and deps, which indicates to select all entries that have records and any dependent entries. See the bib2gls user manual for more details of this option.

### 11.3. Sorting and Displaying the Glossary

With makeindex and xindy, the terms (read from the associated input file) are sorted and the code to typeset the glossary is written to an output file, which is then input by \printglossary. With bib2gls, the entries supplied in the bib files are sorted and the entry definition code (\longnewglossaryentry or \newabbreviation) is written to the glstex file in the order obtained by sorting. This means that the glossary’s internal list is in the required order, so the glossary can be displayed with \printunsrtglossary (see §8.4).

The indexing information, such as the location list or letter groups, is stored in fields such as location or group (where applicable), so the information can be included by \printunsrtglossary, but it means that the information is also available for use elsewhere in the document (so the savenumerlist package option provided by glossaries is redundant).

There are many sorting options provided by bib2gls. The default is to sort according to the system locale. If the document has a language setting, you can use sort=doc to instruct bib2gls to sort according to that. (The language tag obtained from tracklang’s interface is written to the aux file.) For a multilingual document you need to explicitly set the locale using a well-formed language tag. For example:

```latex
\GlsXtrLoadResources[
  src=terms, % data in terms.bib
  sort=de-DE-1996 % sort according to this locale
]```
The locale-sensitive sort methods usually ignore most punctuation so for lists of symbols you may find it more appropriate to use one of the letter-base sort methods that sort according to the Unicode value of each character. Alternatively you can provide a custom rule. See the \texttt{bib2gls} manual for full details of all the available sort methods.

Suppose the \texttt{bib} examples shown earlier have been stored in the files \texttt{terms.bib}, \texttt{abbrvs.bib} and \texttt{symbols.bib} which may either be in the current directory or on \TeX{}’s path. Then the document might look like:

\begin{verbatim}
\documentclass{article}
\usepackage[record]{glossaries-extra}
\setabbreviationstyle{long-short-sc-desc}
\GlsXtrLoadResources[src=terms,abbrvs,symbols]
\begin{document}
\gls{bird}
\gls{shtml}
\gls{M}
\gls{printunsrtglossaries}
\end{document}
\end{verbatim}

The document build process (assuming the document is called \texttt{mydoc}) is:

\begin{verbatim}
pdflatex mydoc
bib2gls mydoc
pdflatex mydoc
\end{verbatim}
11. bib2gls: Managing Reference Databases

Example 151: Using bib2gls: one resource set

bird
server-side includes enabled hypertext markup language (sHTML)
$M$

Glossary

bird feathered animal 1, see also duck & goose

duck a waterbird with short legs

goose a waterbird with a long neck

hypertext markup language (html) a markup language for creating web pages 1

$M$ a matrix 1

server-side includes enabled hypertext markup language (shtml) a combination of hypertext markup language (HTML) and server-side includes (ssi) 1

server-side includes (ssi) a simple interpreted server-side scripting language 1

This creates a single glossary containing the entries: bird, duck, goose, html, M, shtml and ssi (in that order). The bird, shtml and M entries were added because bib2gls detected (from the aux file) that they had been used in the document. The other entries were added because bib2gls detected (from the bib files) that they are referenced by the used entries. In the case of duck and goose, they are in the see field for bird. In the case of ssi and html, they are referenced in the description field of shtml. These cross-referenced entries won’t have a location list when the glossary is first displayed, but depending on how they are referenced, they may pick up a location list on the next document build. The xml entry isn’t required at all, and so hasn’t been defined (from LaTeX’s point of view).

The entries can be separated into different glossaries with different sort methods:

\usepackage[record,abbreviations,symbols]{glossaries-extra}

\setabbreviationstyle{long-short-sc-desc}

\GlsXtrLoadResources[src=terms,sort=en-GB,type=main]

516
\begin{document}
\gls{bird}
\gls{shtml}
\gls{M}
\printunsrtglossaries
\end{document}
Example 152: Using bib2gls: multiple resource sets

\begin{center}
\begin{itemize}
\item bird
\item server-side includes enabled hypertext markup language (SHTML)
\end{itemize}
\end{center}

Glossary

\begin{description}
\item[bird] feathered animal 1, see also duck & goose
\item[duck] a waterbird with short legs
\item[goose] a waterbird with a long neck
\end{description}

Symbols

\begin{description}
\item[M] a matrix 1
\end{description}

Abbreviations

\begin{description}
\item[hypertext markup language (html)] a markup language for creating web pages 1
\item[server-side includes enabled hypertext markup language (shtml)] a combination of hypertext markup language (HTML) and server-side includes (SSI) 1
\item[server-side includes (ssi)] a simple interpreted server-side scripting language 1
\end{description}

Or you can have multiple instance of \GlsXtrLoadResources with the same type, which will produce a glossary with ordered sub-blocks. For example:

\begin{center}
\begin{verbatim}
\usepackage[record,style=indexgroup]{glossaries-extra}
\setabbreviationstyle{long-short-sc-desc}
\GlsXtrLoadResources
[src=abbrvs,sort=letter-nocase,type=main,
group=abbreviations]
\end{verbatim}
\end{center}
This sets the group field for each resource set to the label given by the group resource option. This will result in a glossary where the first group has the label abbreviations and title “Abbreviations”, the second group has the label symbols and title “Symbols” and then follow the usual letter groups. Note that for this example to work, you must run bib2gls with the --group (or -g) switch. For example, if the document is called myDoc.tex:

```bash
pdflatex myDoc
bib2gls -g myDoc
pdflatex myDoc
```
Example 153: Using bib2gls: sub-blocks

bird
server-side includes enabled hypertext markup language (sHTML)
M

Glossary

Abbreviations

hypertext markup language (html) a markup language for creating web pages
server-side includes enabled hypertext markup language (shtml) a combination of hypertext markup language (HTML) and server-side includes (ssi) 1
server-side includes (ssi) a simple interpreted server-side scripting language 1

Abbreviations

M a matrix 1
B
bird feathered animal 1, see also duck & goose
D
duck a waterbird with short legs
G
goose a waterbird with a long neck

The value of the group field must always be a label (so avoid special characters). You can set the corresponding title with glsxtsetgrouptitle (see §8.6). If no title is set then the label is used as the group title.

You can provide your own custom sort rule. For example, if you are using XƎL or LuaƎX:
\GlsXtrLoadResources[
    src={terms}, % entries in terms.bib
    sort=custom, % custom sort rule
    sort-rule={% required with sort=custom
        < æ;Æ < a;á;ä,Ä;Á;Å;Ä < b,B
        < c;ć,C;Ć < d,D < e;é,E;É < f,F < g,G
        < h,H < i;í,I;Í < j,j;J < l,L;Ł < m,M < n,N
        < o;ö;ø,O;Ö;Ø < p,P < q,Q < r,R < s;s,S;Ś < t,T < u;ú,U;Ú < v,V < w,W < x,X < y,Y <
        z;z,Z;Ż
    }
]

With old versions of the \LaTeX kernel, UTF-8 characters, such as é or ø, will expand when written to the aux file.

Some of the options, including \texttt{sort-rule}, allow Unicode characters to be indicated in the format \texttt{\u{\hex}} (or \texttt{\u{\hex}}). \texttt{bib2gls} will recognise this as the character given by the hexadecimal value \texttt{\hex}.

The \texttt{\options} provided to \texttt{\GlsXtrLoadResources} will expand as they are written to the aux file (unless protected). This includes \texttt{\u}, so with a non-Unicode aware engine or where the document source is required to be ASCII, the character æ needs to be written as \texttt{\string\u E6} and so on. Alternatively, use the shortcut \texttt{\string\u E6}.

For example, the above can be rewritten as:

\GlsXtrLoadResources[
    src={terms}, % entries in terms.bib
    sort=custom, % custom sort rule
    sort-rule={% required with sort=custom
        < \glshex E6;\glshex C6
        < a;\glshex E1;\glshex E5,\glshex E4,A;\glshex C1;\glshex C5;\glshex C4
        < b,B < c;\glshex 0107,C;\glshex 0106 < d,D
        < e;\glshex E9,E;\glshex C9 < f,F < g,G
        < h,H < i;\glshex ED,I;\glshex CD < j,J
        < l;\glshex 0142,L;\glshex 0141 < m,M < n,N
    }
]
11.4. Record Counting

As from version 1.1 of bib2gls, you can save the total record count for each entry by invoking bib2gls with the --record-count or --record-count-unit switches. These options will ensure that when each entry is written to the glstex file bib2gls will additionally set the following internal fields for that entry:

- \texttt{recordcount}: set to the total number of records found for the entry;
- \texttt{recordcount.⟨counter⟩}: set to the total number of records found for the entry for the given counter.

If --record-count-unit is used then additionally:

- \texttt{recordcount.⟨counter⟩.⟨location⟩}: set to the total number of records found for the entry for the given counter with the given location.

Only use the unit counting option if the locations don’t contain any special characters. With hyperref use \texttt{\thethe⟨counter-name⟩} rather than \texttt{\thethe⟨counter-name⟩}. Otherwise, if you really need unit counting with locations that may contain formatting commands, then you can try redefining:

\begin{verbatim}
\glsxtrdetoklocation{⟨location⟩}
\end{verbatim}

so that it detokenizes ⟨location⟩ but take care when using \texttt{\GlsXtrLocationRecordCount} with commands like \texttt{\thethepage} as they can end up becoming detokenized too early.

Note that the record count includes locations that bib2gls discards, such as ignored records, duplicates and partial duplicates (unless you filter them out with --record-count-rule). It doesn’t include cross-reference records. For example, suppose a document has an entry with the label \texttt{bird} that is recorded (indexed) as follows:

\begin{itemize}
  \item \textbf{Page 1} two (2) instances of \texttt{\gls{bird}};
  \item \textbf{Page 2} one (1) instance of \texttt{\gls{bird}};
  \item \textbf{Page 3} four (4) instances of \texttt{\gls{bird}};
\end{itemize}
Section 3  one (1) instance of \gls{bird}.

Then the total record count (stored in the \texttt{recordcount} field) is $2 + 1 + 4 + 1 = 8$, the total for the page counter (stored in the \texttt{recordcount.page} field) is $2 + 1 + 4 = 7$, and the total for the section counter (stored in the \texttt{recordcount.section} field) is 1.

With the unit counting on as well, the following fields are assigned:

- \texttt{recordcount.page.1} is set to 2;
- \texttt{recordcount.page.2} is set to 1;
- \texttt{recordcount.page.3} is set to 4;
- \texttt{recordcount.section.3} is set to 1.

You can access these fields using the following commands which will expand to the field value if set or to 0 if unset:

\begin{itemize}
\item \texttt{\GlsXtrTotalRecordCount{⟨entry-label⟩}}
\item \texttt{\GlsXtrRecordCount{⟨entry-label⟩}{⟨counter⟩}}
\item \texttt{\GlsXtrLocationRecordCount{⟨entry-label⟩}{⟨counter⟩}{⟨location⟩}}
\end{itemize}

This expands to the total record count for the entry given by \texttt{⟨label⟩}. For example:

\begin{itemize}
\item \texttt{\GlsXtrTotalRecordCount{bird}} expands to 8.
\item \texttt{\GlsXtrRecordCount{bird}{page}} expands to 7 and
\item \texttt{\GlsXtrRecordCount{bird}{section}} expands to 1.
\end{itemize}

\texttt{\GlsXtrLocationRecordCount{⟨entry-label⟩}{⟨counter⟩}{⟨location⟩}}
This expands to the total for the given location. For example

\texttt{\GlsXtrLocationRecordCount\{bird\}\{page\}\{3\}}

expands to 4. Be careful about using \texttt{\thepage} in the \langle location \rangle part. Remember that due to \TeX{}'s asynchronous output routine, \texttt{\thepage} may not be correct.

There are commands analogous to the entry counting commands like \texttt{\cgls} and \texttt{\cgls-format} that are triggered by the record count. These are listed below. The test to determine if the entry’s record count exceeds the trigger value (which should be stored in the record-count attribute) is obtained with:

\texttt{\glsxtrifrecordtrigger\{⟨entry-label⟩\}\{(true)\}\{(false)\}}

If the recordcount attribute is set and ⟨total⟩ exceeds the value given by the recordcount attribute, then this does ⟨true⟩ otherwise it does ⟨false⟩. The ⟨total⟩ is given by:

\texttt{\glsxtrrecordtriggervalue\{⟨entry-label⟩\}}

This should expand to the record count value that needs testing. The default definition is:

\texttt{\newcommand*{\glsxtrrecordtriggervalue}\[1\]{%}
 \GlsXtrTotalRecordCount\{#1\}%
 }

This command may be redefined as appropriate. For example, it may be redefined to use \texttt{\GlsXtrRecordCount} for a particular location counter or to use \texttt{\GlsXtrLocationRecordCount} for a particular location.

The recordcount attribute may be set with \texttt{\glssetcategoryattribute} or can be set for each listed category with:

\texttt{\GlsXtrSetRecordCountAttribute\{⟨category-list⟩\}\{⟨value⟩\}}

The value must be an integer.

Commands like \texttt{\rgls} behave slightly differently to \texttt{\cgls}. It’s necessary for the command to add a record to the aux file in order for the entry to be selected and for the record count to be correct on the next \texttt{\bib2gls+\LaTeX} run (for the default selection=\{recorded and deps\}). The trigger record is created with format=glstriggerrecordformat, which \texttt{\bib2gls v1.1+} recognises as a special type of ignored location format. This corresponds to

524
the command:

\texttt{\glstriggerrecordformat{(location)}}

As with \texttt{\glsignore}, this command does nothing and is considered an ignored record (so it won't appear in the location list), but it indicates to \texttt{bib2gls} that the entry must be selected and, if the \texttt{trigger-type} option has been set, the entry will be assigned to the \texttt{trigger-type} glossary.

For example, to assign the entry to an ignored glossary:

\texttt{\newignoredglossary{ignored}}  
\texttt{\GlsXtrLoadResources[trigger-type=ignored]}

This ensures that the entry is defined but it won’t show up the normal glossary.

The post-link hook won’t be implemented if the record trigger is tripped. (That is, if the \texttt{\rglsformat-like command is used instead of the \gls-like command.})

\texttt{\rgls[(options)]{(entry-label)}{(insert)} modifiers: \ast \langle alt-mod \rangle}

If the value has been supplied by the \texttt{recordcount} attribute and is exceeded, this behaves like \texttt{\gls} otherwise it creates a trigger record and uses:

\texttt{\rglsformat{(entry-label)}{(insert)}}

This has the same definition as \texttt{\cglsformat}.

\texttt{\rglsp1[(options)]{(entry-label)}{(insert)} modifiers: \ast \langle alt-mod \rangle}

If the value has been supplied by the \texttt{recordcount} attribute and is exceeded, this behaves like \texttt{\glspl} otherwise it creates a trigger record and uses:

\texttt{\rglsp1format{(entry-label)}{(insert)}}
which uses the appropriate plural fields.

\rGlss[{options}]{⟨entry-label⟩}{⟨insert⟩}

\rGlssformat{⟨entry-label⟩}{⟨insert⟩}

which performs the appropriate case-change.

\rGlsp1[{options}]{⟨entry-label⟩}{⟨insert⟩}

\rGlsp1format{⟨entry-label⟩}{⟨insert⟩}

which uses the appropriate plural fields and case-change.

\rGLS[{options}]{⟨entry-label⟩}{⟨insert⟩}

\rGLSformat{⟨entry-label⟩}{⟨insert⟩}

which performs the appropriate case-change.

\rGLSp1[{options}]{⟨entry-label⟩}{⟨insert⟩}

\rGLSp1format{⟨entry-label⟩}{⟨insert⟩}

If the value has been supplied by the `recordcount` attribute and is exceeded, this behaves like \rGlss otherwise it creates a trigger record and uses:
which uses the appropriate plural fields and case-change.
To make it easier to switch on record counting for an existing document, you can use:

\glsxtrenablerecordcount

This redefines \gls, \glspl, \Gls, \GPLS, \GLSP to \rgls, \rglspl, \rGls, \rGLS- pl, \rGLS, \rGLSP, respectively, for convenience. This command will also switch the shortcut commands such as \ac or \ab, if they have been enabled, from using the \cglst-like commands to the corresponding \rgls command.

For example, using the earlier terms.bib, abbrvs.bib and symbols.bib example files:

\documentclass{article}
\usepackage[record]{glossaries-extra}
\newignoredglossary{ignored}
\setabbreviationstyle{long-short-sc-desc}
\GlsXtrLoadResources[  
src={terms,abbrvs,symbols},
  trigger-type=ignored,
  category={same as entry}
]
\glsxtrenablerecordcount
\GlsXtrSetRecordCountAttribute{general,abbreviation}{1}
\glsdefpostlink{entry}{\glsxtrpostlinkAddDescOnFirstUse}
\glsdefpostlink{symbol}{\glsxtrpostlinkAddDescOnFirstUse}
\begin{document}
\gls{bird}, \gls{ssi}, \gls{bird}, \gls{html}, \gls{M}, \gls{html}.
\printunsrtglossaries
\end{document}

If the document is called myDoc.tex, then the build process is:
The `category={same as entry}` resource option assigns the `category` field to the `bib` entry type (without the initial @). This means that the entries defined in `terms.bib` (with @entry) have their `category` set to `entry`, the entries defined in `abbrvs.bib` (with @abbreviation) have their `category` set to `abbreviation`, and the entries defined in `symbols.bib` (with @symbol) have their `category` set to `symbol`.

I’ve added post-link hooks to the `entry` and `symbol` categories to show the description on first use (but not for the `abbreviation` category).

Example 154: Using `bib2gls`: record counting

\[ M \quad \text{(a matrix)}, \quad \text{server-side includes,} \quad \text{bird (feathered animal),} \quad \text{hypertext markup language (HTML),} \quad M, \quad \text{HTML}. \]

**Glossary**

- **bird**: feathered animal 1, *see also* duck & goose
- **duck**: a waterbird with short legs
- **goose**: a waterbird with a long neck
- **hypertext markup language (html)**: a markup language for creating web pages 1
- **M**: a matrix 1

In the above, `ssi` and `bird` only have one record. However, they have been treated differently. The `ssi` entry is using \glsformat{} whereas the `bird` entry is using the normal \gls{} behaviour. This is because the record counting hasn’t been applied to the custom `entry` category, whereas it has been applied to the `abbreviation` and `symbol` categories.

**11.4.1. Unit Record Counting**

If you want unit record counting you need to remember to invoke `bib2gls` with `--record-count-unit` and you will also need to redefine \glsxtrrecordtrigger{} appropriately. For example, suppose you want to reset all abbreviations at the start of each chapter, so that the full form is shown again, but only if the abbreviation isn’t used elsewhere in the chapter.
This would require records with the counter set to chapter. This can be done with the counter package option:

\usepackage[record,counter=chapter]{glossaries-extra}

However, this will have chapter numbers instead of page numbers in the location lists. If you don’t want location lists then this isn’t a problem. The list can simply be suppressed with nonumberlist.

If you want page numbers in the location lists then you will need to record each entry with both the page and chapter counters. This can be done with the hook that occurs before the \gls options are set:

\renewcommand{\glslinkpresetkeys}{%\glsadd[format=glsignore,counter=chapter]{\glslabel}}

Note that I’ve used the ignored location format to prevent the chapter number from being added to the location list. An alternative is to use the loc-counters=page resource option to only show the locations that use the page counter.

The definition of \glsxtrrecordtriggervalue needs to be changed so that it uses the total for the given location. If hyperref is used, you will need \theHchapter:

\renewcommand*{\glsxtrrecordtriggervalue}[1]{%\GlsXtrLocationRecordCount{#1}{chapter}{\theHchapter} %}

otherwise use \thechapter.

Consider the following (using the abbreviations defined in the earlier abbrvs.bib):

\begin{document}
\chapter{First}
\gls{html}. \gls{html}. \gls{html}. \gls{ssi}.
\chapter{Second}
\gls{html}. \gls{ssi}. \gls{ssi}.
\gls{xml}.
\printunsrtglossaries
\end{document}

Note that the xml entry is only used once in the entire document, but it will still be added to
the glossary.

The previous example used the `trigger-type` resource option to move entries with the `glstriggerrecordformat` encap (that is, they didn’t exceed the trigger value) to another glossary. Unfortunately, using that option in this case will move all three abbreviations to the `trigger-type` glossary. The `ssi` entry is only used once in the first chapter (but is used twice in the second chapter), and the `html` is only used once in the second chapter (but is used three times in the first chapter). So all three will have records in the aux file with the special `glstriggerrecordformat` format.

A simple solution is to omit any entries that don’t have the `location` field set when displaying the glossary:

```latex
\renewcommand*{\printunsrtglossaryentryprocesshook}[1]{% 
  \glsxtrifhasfield*{location}{#1}{%}
  \printunsrtglossaryskipentry%
\}
\printunsrtglossaries
```

An alternative is to test the total record count, but remember that each entry is being recorded twice: once with the page counter and once with the chapter counter, so the total count for the `ssi` entry will be 2 not 1.

```
\documentclass{scrreport}
\usepackage[T1]{fontenc}
\usepackage[colorlinks]{hyperref}
\usepackage[record,postdot]{glossaries-extra}
\setabbreviationstyle{long-short-sc-desc}
\GlsXtrLoadResources[src={abbrvs}]
\preto{\chapter}{\glsresetall}
\glsxtrenablerecordcount
\GlsXtrSetRecordCountAttribute{abbreviation}{1}
\renewcommand{\glslinkpresetkeys}{% 
```

Take care not to strip entries from a hierarchical glossary as it will break the hierarchy and will cause formatting problems in the glossary.

The complete document is:

```latex
\documentclass{scrreport}
\usepackage[T1]{fontenc}
\usepackage[colorlinks]{hyperref}
\usepackage[record,postdot]{glossaries-extra}
\setabbreviationstyle{long-short-sc-desc}
\GlsXtrLoadResources[src={abbrvs}]
\preto{\chapter}{\glsresetall}
\glsxtrenablerecordcount
\GlsXtrSetRecordCountAttribute{abbreviation}{1}
\renewcommand{\glslinkpresetkeys}{% 
```
11. \textit{bib2gls}: Managing Reference Databases

\begin{verbatim}
\glsadd[format=glsignore,counter=chapter]{\glslabel}
\renewcommand*{\glsxtrrecordtriggervalue}[1]{
  \GlsXtrLocationRecordCount{#1}{chapter}{\thechapter}
}
\begin{document}
\chapter{First}
gls{html}. \gls{html}. \gls{html}. \gls{ssi}.
\chapter{Second}
gls{html}. \gls{ssi}. \gls{ssi}. \gls{xml}.
\renewcommand*{\printunsrtglossaryentryprocesshook}[1]{
  \glsxtrifhasfield*{location}{#1}
  {}\{\printunsrtglossaryskipentry\%
}
\printunsrtglossaries
\end{document}
\end{verbatim}

If the document is in a file called \texttt{myDoc.tex} then the document build is:

\begin{verbatim}
pdflatex myDoc
bib2gls --record-count-unit myDoc
pdflatex myDoc
\end{verbatim}

\begin{example}
Example 155: Using \textit{bib2gls}: unit record counting
\begin{tabular}{|c|c|c|}
\hline
\textbf{1 First} & \textbf{2 Second} & \textbf{Glossary} \\
\hline
hypertext markup language (html) & server-side includes (ssi) & \textit{html} \textit{html} \textit{html} \textit{html} \\
\hline
\end{tabular}
\end{example}

11.4.2. Mini-Glossaries

Record counting doesn’t have to be used with the \texttt{rgls} set of commands. When \textit{bib2gls} writes the code to the \texttt{glstex} file to save the record counting information, it does it with helper commands that it provides in the \texttt{glstex} file:

\begin{verbatim}
\bibglssettotalrecordcount{⟨entry-label⟩}{⟨value⟩}
\end{verbatim}

This sets the total record count and is defined in the \texttt{glstex} file as:
This sets the total for the given counter and is defined as:

\providecommand*\bibglssetrecordcount[3]{% 
\GlsXtrSetField{#1}{recordcount.#2}{#3}%
}\bibglssetrecordcount{⟨entry-label⟩}{⟨counter⟩}{⟨value⟩}

The following is only available with \--record-count-unit:

\providecommand*\bibglssetlocationrecordcount[4]{% 
\GlsXtrSetField{#1}{recordcount.#2.#3}{#4}\
\provideignoredglossary{minigloss.#2.#3}\glsxtrcopytoglossary{#1}minigloss.#2.#3%
}\bibglssetlocationrecordcount{⟨entry-label⟩}{⟨counter⟩}{⟨location⟩}{⟨value⟩}

This sets the total for the given location and is defined as:

\providecommand*\bibglssetlocationrecordcount[4]{% 
\GlsXtrSetField{#1}{recordcount.#2.#3}{#4}\
\provideignoredglossary{minigloss.#2.#3}\glsxtrcopytoglossary{#1}minigloss.#2.#3%
}\bibglssetlocationrecordcount{⟨entry-label⟩}{⟨counter⟩}{⟨location⟩}{⟨value⟩}

By defining one of more of these commands before the \glistex file is input, it’s possible to pick up the information, without the need to iterate over all entries later. For example, the following will create a mini-glossary for each particular location and populate it with entries that have a record for that location.

\newcommand*\bibglssetlocationrecordcount[4]{% 
\GlsXtrSetField{#1}{recordcount.#2.#3}{#4}\
\provideignoredglossary{minigloss.#2.#3}\glsxtrcopytoglossary{#1}minigloss.#2.#3%
}\bibglssetlocationrecordcount{⟨entry-label⟩}{⟨counter⟩}{⟨location⟩}{⟨value⟩}

I’ve omitted \glsxtrdetoklocation for clarity and because I’m confident the locations won’t be problematic. The mini-glossary can then be displayed at the start of the chapter with:
The previous example can be altered to strip the \rgls commands and instead add a mini-glossary at the start of each chapter (the redefinition of \glslinkpresetkeys remains to ensure there are locations with the chapter counter). I’ve also provided a command to make it easier to display the mini-glossaries:

```latex
\newcommand{\minigloss}{%  
    \printunsrtglossary*[style=abbr-short-long, type=minigloss.chapter.\theHchapter, groups=false, target=false]{2}  
    {\renewcommand{\glossarysection}[2][]{}  
    \renewcommand{\glslongextraShortLongTabularHeader}{\toprule}  
}}
```

The document build is the same.

Example 156: Using bib2gls: unit record counting mini-glossary

<table>
<thead>
<tr>
<th>1 First</th>
<th>2 Second</th>
<th>Glossary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>hypertext markup language (html) a markup language for creating web pages. 1, 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>server-side includes (ssi) a simple interpreted server-side scripting language. 1, 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>extensible markup language (xml) a simple text-base format for representing structured information. 2</td>
</tr>
</tbody>
</table>

11.5. The glossaries-extra-bib2gls package

```latex
\usepackage{glossaries-extra-bib2gls}  
```

The package options record=only (or simply record) and record=nameref automatically loads the supplementary package glossaries-extra-bib2gls, which provides some commands that are specific to bib2gls, in particular to sorting and location lists which aren’t relevant with record=hybrid.

If glossaries-extra-bib2gls is loaded via the record package option then the check for associated language files (see §15) will also search for the existence of glossariesextr-⟨script⟩.1df for each document dialect (where ⟨script⟩ is the four letter script identifier, such as Latn).
11.5.1. Displaying Glossaries

Glossaries are displayed with the "unsrt" family of commands (see §8.4). Some styles, such as bookindex, are customized for use with bib2gls.

The following commands are shortcuts that use \printunsrtglossary. However, they are only defined if a corresponding package option has been set before glossaries-extra-bib2gls is loaded. This means that the options must be passed as a package option, not using \glossariesextrasetup, if the shortcut commands are required.

\printunsrtabbreviations[⟨options⟩]

This shortcut is provided if the abbreviations package option has been used. This is a shortcut for:

\printunsrtglossary[type=abbreviations,⟨options⟩]

\printunsrtacronyms[⟨options⟩]

This shortcut is provided if the acronyms (or acronym) package option has been used. This is a shortcut for:

\printunsrtglossary[type=\acronymtype,⟨options⟩]

\printunsrtsymbols[⟨options⟩]

This shortcut is provided if the symbols package option has been used. This is a shortcut for:

\printunsrtglossary[type=symbols,⟨options⟩]

\printunsrtnumbers[⟨options⟩]

This shortcut is provided if the numbers package option has been used. This is a shortcut for:
11.5.2. Helper Commands for Resource Options

\glshex\langle hex\rangle

This simply expands to \texttt{\string\u\langle hex\rangle}, which is used to identify the Unicode character \langle hex\rangle in the value of some resource options.

\glshashchar

This expands to a literal # character (similar to \glshash).

\glscapturedgroup\langle n\rangle

This simply expands to \texttt{\string\$\langle n\rangle} which is used to indicate the \langle n\rangle th captured group in a regular expression replacement in the value of some resource options (requires bib2gls v1.5+), such as \texttt{sort-replace}. For example:

\texttt{sort-replace={{{[a-zA-Z]}\string\}.}{\glscapturedgroup1}}

This removes a full stop that follows any of the characters a,...,z or A,...,Z. A convenient shortcut for use in the \texttt{entry-type-aliases} setting:

\texttt{\GlsXtrBibTeXEntryAliases}

535
This provides aliases for \LaTeX’s standard entry types (such as \texttt{@article} and \texttt{@book}) to \bib2gls’s \texttt{@bibtexentry} entry type (requires \bib2gls v1.4+).

You may also want to provide storage keys for \LaTeX’s standard fields rather than having to alias them all. This can be done with:

\begin{verbatim}
\GlsXtrProvideBibTeXFields
\end{verbatim}

This defines each \LaTeX field, such as \texttt{author}, as a glossary entry key:

\begin{verbatim}
\glsaddstoragekey{address}{}{\glsxtrbibaddress}\
\glsaddstoragekey{author}{}{\glsxtrbibauthor}\
\glsaddstoragekey{booktitle}{}{\glsxtrbibbooktitle}\
\glsaddstoragekey{chapter}{}{\glsxtrbibchapter}\
\glsaddstoragekey{edition}{}{\glsxtrbibedition}\
\glsaddstoragekey{howpublished}{}{\glsxtrbibhowpublished}\
\glsaddstoragekey{institution}{}{\glsxtrbibinstitution}\
\glsaddstoragekey{journal}{}{\glsxtrbibjournal}\
\glsaddstoragekey{month}{}{\glsxtrbibmonth}\
\glsaddstoragekey{note}{}{\glsxtrbibnote}\
\glsaddstoragekey{number}{}{\glsxtrbibnumber}\
\glsaddstoragekey{organization}{}{\glsxtrbiborganization}\
\glsaddstoragekey{pages}{}{\glsxtrbibpages}\
\glsaddstoragekey{publisher}{}{\glsxtrbipublisher}\
\glsaddstoragekey{school}{}{\glsxtrbibschool}\
\glsaddstoragekey{series}{}{\glsxtrbibseries}\
\glsaddstoragekey{title}{}{\glsxtrbibtitle}\
\glsaddstoragekey{bibtextype}{}{\glsxtrbibtype}\
\glsaddstoragekey{volume}{}{\glsxtrbibvolume}
\end{verbatim}

This command should be placed before the first \texttt{\GlsXtrLoadResources}.

\textbf{\LaTeX’s \texttt{type} field clashes with the glossaries package’s \texttt{type} key, so this command provides the key \texttt{bibtextype} instead. You can alias it with \texttt{field-aliases={type=bibtextype}} in the resource options.}

\section*{11.5.2.1. Custom Sort}

There are many locale alphabetical rules provided with \bib2gls, such as \texttt{sort=de-1996} for German new orthography. However, it may be that your particular locale isn’t supported, or you want a rule that covers multiple scripts or non-alphabetic symbols.

The \texttt{sort=custom} setting combined with \texttt{sort-rule} provides a way to define your own sort rule. For example, suppose I have a file called \texttt{animals.bib} that contains:
Here's a very limited rule that only recognises five letters:

\usepackage[record,nostyles,stylemods=bookindex,style=bookindex]{glossaries-extra}
\newcommand{\bibglssetlastgrouptitle}{2}{% 
\glsxtrsetgrouptitle{#1#2}{Other}%
}\GlsXtrLoadResources[src=animals,selection=all, sort=custom,sort-rule={ < a,A < b,B < e,E < l,L < ll,Ll,LL < z,Z}] 
\begin{document} 
\printunsrtglossaries 
\end{document} 

Any characters that aren't included in the rule (such as “c” and “g”) are placed at the end. I've defined \bibglssetlastgrouptitle to label that final group of characters “Other”. If the document is in a file called myDoc.tex, the build process is:

pdflatex myDoc 
bib2gls -g myDoc 
pdflatex myDoc

The result is:
Example 157: Using \texttt{bib2gls}: simple custom sort rule

\begin{table}
\centering
\begin{tabular}{l l}
\hline
A & lynx \\
\midrule
ant & \\
\hline
B & llama \\
\midrule
bat & \\
bee & \\
\hline
E & zebu \\
\midrule
elk & \\
egret & \\
\hline
L & cow \\
\midrule
lion & goose \\
\hline
\end{tabular}
\end{table}

Note that “egret” has been placed after “elk”. This is because “l” is included in the rule but “g” isn’t. Whereas “lynx” comes before “llama” because there’s a separate “ll” group after the “l” group.

The commands listed below provide common rule blocks for use in the \texttt{sort-rule} resource option. If you want a rule for a specific locale, you can provide similar commands in a file called \texttt{glossariesxtr-⟨tag⟩.ldf}, where ⟨tag⟩ identifies the dialect, locale, region or root language. See the description of \texttt{IfTrackedLanguageFileExists} in the tracklang documentation for further details. If this file is on \TeX’s path and the tracklang package (automatically loaded by glossaries) detects that the document has requested that language or locale, then the file will automatically be loaded. For example, if you want to provide a rule block for Welsh, then create a file called \texttt{glossariesxtr-welsh.ldf} that contains:

\begin{verbatim}
\ProvidesGlossariesExtraLang{welsh}[2018/02/23 v1.0]
\@ifpackageloaded{glossaries-extra-bib2gls}
{
  \newcommand{\glsxtrWelshRules}{%
    \glsxtrLatinA
    \string<b,B
    \string<c,C
    \string<ch,CH
\%}
\end{verbatim}
(The use of \string is in case the < character has been made active.) You can provide more than one rule-block per local, to allow for loanwords or foreign words. For example, you could provide \glsxtrWelshIRules, \glsxtrWelshIIRules etc.

If the rules are for a particular script (independent of language or region) then they can be provided in a file given by glossariesxtr-⟨script⟩.ldf instead. For example, the file glossariesxtr-Cyrl.1df could contain:

\ProvidesGlossariesExtraLang{Cyrl}[2018/02/23 v1.0]
\newcommand*{\glsxtrGeneralCyrillicIRules}{\%
  \% Cyrillic rules
  }
\newcommand*{\glsxtrGeneralCyrillicIIRules}{\%
Remember that the required document language scripts need to be tracked through the track-lang package, in order for these files to be automatically loaded. This essentially means ensuring you load the appropriate language package before tracklang is loaded by the base glossaries package or any other package that uses it. See the tracklang documentation for further details.

Alternatively, if the rules are specific to a subject rather than a region or language, then you can provide a supplementary package. For example, if you have a package called, say, mapsymbols that provides map symbols, then the file mapsymbols.sty might contain:

\ProvidesPackage{mapsymbols}
% some package or font loading stuff here to provide
% the appropriate symbols
\newcommand{\Stadium}{...}
\newcommand{\Battlefield}{...}
\newcommand{\Harbour}{...}
% etc

% Provide a rule block:
\newcommand{\MapSymbolOrder}{%
  \glshex 2694 % crossed-swords 0x2694
  \string< \glshex 2693 % anchor 0x2693
  \string< \glshex 26BD % football 0x26BD
}  

and the supplementary file mapsymbols.bib can provide the appropriate definitions for bib2gls:

@preamble{"\glsxtrprovidecommand{\Harbour}{\char"2693}
\glsxtrprovidecommand{\Battlefield}{\char"2694}
\glsxtrprovidecommand{\Stadium}{\char"26BD}"
}

Now both the preamble and rule block can be used in the resource set:

\usepackage{mapsymbols}% my custom package
\usepackage[record]{glossaries-extra}
\GlsXtrLoadResources[
(As before, you may need to use `\string` in front of characters like `<` if they have been made active.)

The following commands are provided by `glossaries-extra-bib2gls`. They should be separated by the rule separator characters `;` (semi-colon) or `,` (comma) or `&` (ampersand) or `<` (less than). See Java’s `RuleBasedCollator` documentation for details of the rule syntax.

For example, the following will place the mathematical Greek symbols (\alpha, \Alpha, \beta, \Beta etc) in a block before Latin characters:

```latex
sort-rule={\glsxtrcontrolrules ;\glsxtrspacerules ;\glsxtrnonprintablerules ;\glsxtrcombiningdiacriticrules ,\glsxtrhyphenrules <\glsxtrgeneralpuncrules <\glsxtrdigitrules <\glsxtrfractionrules <\MapSymbolOrder <\glsxtrMathItalicGreekIrules <\glsxtrGeneralLatinIVrules <\glsxtrLatinAA <\glsxtrLatinOslash }
```
11.5.2.1.1. Non-Letters

\texttt{\textbackslash glsxtrcontrolrules}

These are control characters that are usually placed at the start of a rule in the ignored section. They typically won’t occur in any sort values, but if they do they should normally be ignored.

\texttt{\textbackslash glsxtrspacerules}

These are space characters. They typically come after the control characters with the two blocks separated by a \texttt{;} (semi-colon).

\texttt{\textbackslash glsxtrnonprintablerules}

These are non-printable characters (BOM, tabs, line feed and carriage return). They typically come after the spaces separated by a \texttt{;} (semi-colon). These characters aren’t checked for by \texttt{bib2gls} when it determines whether or not to use the interpreter, so a TAB or newline character may end up in the sort value if it wasn’t interpreted.

\texttt{\textbackslash glsxtrcombiningdiacriticrules}

These are combining diacritic marks which typically follow the space and non-printable blocks (separated by a semi-colon). This command is defined in terms of sub-block commands:

\begin{verbatim}
\newcommand*{\glslxtrcombiningdiacriticrules}{% \\
\glslxtrcombiningdiacriticIrules\string; \\
\glslxtrcombiningdiacriticIIrules\string; \\
\glslxtrcombiningdiacriticIIIrules\string; \\
\glslxtrcombiningdiacriticIVrules
\}
\end{verbatim}

If you prefer, you can use the sub-blocks directly in your required ordered.

\texttt{\textbackslash glsxtrcombiningdiacriticIrules}

This contains the combining diacritics: acute, grave, breve, circumflex, caron, ring, vertical
line above, diaeresis (umlaut), double acute, tilde, dot above, combining macron.

\texttt{\glsxtrcombiningdiacriticIIrules}

This contains the combining diacritics: short solidus overlay, cedilla, ogonek, dot below, low line, overline, hook above, double vertical line above, double grave accent, candrabindu, inverted breve, turned comma above, comma above, reversed comma above, comma above right, grave accent below, acute accent below.

\texttt{\glsxtrcombiningdiacriticIIrules}

This contains the combining diacritics: left tack below, right tack below, left angle above, horn, left half ring below, up tack below, down tack below, plus sign below, minus sign below, palatalized hook below, retroflex hook below, diaeresis below, ring below, comma below, vertical line below, bridge below, inverted double arch below, caron below, circumflex accent below, breve below, inverted breve below, tilde below, macron below, double low line, tilde overlay, short stroke overlay, long stroke overlay, long solidus overlay, right half ring below, inverted bridge below, square below, seagull below, x above, vertical tilde, double overline, Greek perisopmeni, Greek dialytika tonos, Greek ypogeogrammeni, double tilde, double inverted breve, Cyrillic titlo, Cyrillic palatalization, Cyrillic dasia pneumata, Cyrillic psili pneumata.

\texttt{\glsxtrcombiningdiacriticIVrules}

This contains the combining diacritics: left harpoon above, right harpoon above, long vertical line overlay, short vertical line overlay, anticlockwise arrow above, clockwise arrow above, left arrow above, right arrow above, ring overlay, clockwise ring overlay, anticlockwise ring overlay, three dots above, four dots above, enclosing circle, enclosing square, enclosing diamond, enclosing circle backslash, left right arrow above.

\texttt{\glsxtrhyphenrules}

This contains hyphens (including the minus sign 0x2212). This rule block typically comes after the diacritic rules separated by a comma.

\texttt{\glsxtrgeneralpuncrules}

This contains punctuation characters. This rule block typically comes after the hyphen rules separated by a less than (<). As with the combining diacritics, this command is defined in
11. bib2gls: Managing Reference Databases

terms of sub-blocks which may be used directly instead if a different order is required:

\newcommand*{\glsxtrgeneralpuncrules}{%  
\glsxtrgeneralpuncIrules  \string<\glsxtrcurrencyrules  \string<\glsxtrgeneralpuncIIrules
}  

\glsxtrgeneralpuncIrules

This is defined as:

\newcommand*{\glsxtrgeneralpuncIrules}{%  
\glsxtrgeneralpunchmarksrules  \string<\glsxtrgeneralpuncaccentsrules  \string<\glsxtrgeneralpuncquoterules  \string<\glsxtrgeneralpuncbracketrules  \string<\glsxtrgeneralpuncsignrules
}  

\glsxtrgeneralpunchmarksrules

This contains: underscore, macron, comma, semi-colon, colon, exclamation mark, inverted exclamation mark, question mark, inverted question mark, solidus, full stop.

\glsxtrgeneralpuncaccentsrules

This contains: acute accent, grave accent, circumflex accent, diaeresis, tilde, middle dot, cedilla.

\glsxtrgeneralpuncquoterules

This contains: straight apostrophe, straight double quote, left guillemet, right guillemet.

\glsxtrgeneralpuncbracketrules
This contains: left parenthesis, right parenthesis, left square bracket, right square bracket, left curly bracket, right curly bracket.

\glsxtrgeneralpuncsignrules

This contains: section sign, pilcrow sign, copyright sign, registered sign, at sign.

\glsxtrcurrencyrules

This sub-block contains some currency symbols: currency sign, Thai currency symbol baht, cent sign, colon sign, cruzeiro sign, dollar sign, dong sign, euro sign, French franc sign, lira sign, mill sign, naira sign, peseta sign, pound sign, rupee sign, new sheqel sign, won sign, yen sign.

\glsxtrgeneralpuncIIrules

This sub-block contains some other punctuation symbols: asterisk, backslash, ampersand, hash sign, percent sign, plus sign, plus-minus sign, division sign, multiplication sign, less-than sign, equals sign, greater-than sign, not sign, vertical bar (pipe), broken bar, degree sign, micron sign.

\glsxtrdigitrules

This rule block contains the Basic Latin digits (0, ..., 9) and the subscript and superscript digits (\(\alpha^0\) etc) made equivalent to the corresponding Basic Latin digit. The digit block typically comes after the punctuation rules separated by a less than (<).

\glsxtrBasicDigitrules

This rule block contains just the Basic Latin digits (0, ..., 9).

\glsxtrSubScriptDigitrules

This rule block contains just the subscript digits (\(0 \ldots 9\)).

\glsxtrSuperScriptDigitrules
This rule block contains just the superscript digits ($^0 \ldots ^9$).

\glsxtrfractionrules

This rule block contains vulgar fraction characters from the Unicode Number Forms block. The digit block typically comes after the digit rules separated by a less than ($<$).

\glsxtrIgnorableRules

A shortcut that expands to the ignorable rules:

\glsxtrcontrolrules
;\glsxtrspacerules
;\glsxtrnonprintablerules

\glsxtrGeneralInitRules

A shortcut that expands to common initial rules:

\glsxtrIgnorableRules
;\glsxtrcombiningdiacriticrules
;\glsxtrhyphenrules
<$\glsxtrgeneralpuncrules
<$\glsxtrdigitrules
<$\glsxtrfractionrules

Note that this includes the combining diacritic rules, which won’t be appropriate for languages with accented characters.

There are a number of Latin rule blocks. Some of these included extended characters or ligatures (such as ß or œ) but they don’t include accented characters. If you require a Latin rule block that includes accented characters, digraphs, trigraphs or other extended characters, then it’s best to provide similar commands in a *glossariesxtr-⟨tag⟩.ldf* file for the particular language or region.

\textbf{11.5.2.1.2. Latin Letters}

\glsxtrGeneralLatinIrules

This is just the basic (non-extended) Latin alphabet with the superscript and subscript Latin letters ($^a \ldots ^z$, etc) treated as the equivalent basic Latin letter. (If you don’t want the subscripts
and superscripts included you can redefine \glsxtrLatinA etc to omit them.)

\glsxtrGeneralLatinIIrules

This is like \glsxtrGeneralLatinIrules but it includes eth (Ð) between “D” and “E” and eszett (ß) treated as “ss”.

\glsxtrGeneralLatinIIIrules

This is like \glsxtrGeneralLatinIrules but it includes eth (Ð) between “D” and “E” and eszett (ß) treated as “sz”.

\glsxtrGeneralLatinIVrules

This is like \glsxtrGeneralLatinIrules but it includes eth (Ð) between “D” and “E”, ae- ligature (æ) is treated as “ae”, oe-ligature (œ) is treated as “oe”, eszett (ß) treated as “ss” and thorn (þ) is treated as “th”.

\glsxtrGeneralLatinVrules

This is like \glsxtrGeneralLatinIrules but it includes eth (Ð) between “D” and “E”, eszett (ß) treated as “ss” and thorn (þ) is treated as “th”.

\glsxtrGeneralLatinVIrules

This is like \glsxtrGeneralLatinIrules but it includes eth (Ð) between “D” and “E”, eszett (ß) treated as “sz” and thorn (þ) is treated as “th”.

\glsxtrGeneralLatinVIIrules

This is like \glsxtrGeneralLatinIrules but it includes ae-ligature (æ) between “A” and “B”, eth (Ð) between “D” and “E”, insular G (ᵹ) instead of “G”, oe-ligature (œ) between “O” and “P”, long s (ſ) equivalent to “s”, thorn (þ) between “T” and “U” and wynn (ƿ) instead of “W”.

\glsxtrGeneralLatinVIIIrules

547
This is like \glsxtrGeneralLatinIrules but ae-ligature (æ) is treated as “ae”, oe-ligature (œ) is treated as “oe”, eszett (ß) treated as “ss”, thorn (þ) is treated as “th”, Ø is treated as “O” and “Ł” is treated as “L”.

\glsxtrGeneralLatinAtoMrules
A mini-rule subset of General Latin I rules that just covers A–M.

\glsxtrGeneralLatinNtoZrules
A mini-rule subset of General Latin I rules that just covers N–Z.

\glsxtrGeneralLatinAtoGrules
A mini-rule subset of General Latin I rules that just covers A–G.

\glsxtrGeneralLatinHtoMrules
A mini-rule subset of General Latin I rules that just covers H–M.

\glsxtrGeneralLatinNtoSrules
A mini-rule subset of General Latin I rules that just covers N–S.

\glsxtrGeneralLatinTtoZrules
A mini-rule subset of General Latin I rules that just covers T–Z.

\glsxtrLatinA
A mini-rule that just covers “A” but includes the sub- and superscript A.

\glsxtrLatinE
A mini-rule that just covers “E” but includes the subscript E.

\glsxtrLatinH

A mini-rule that just covers “H” but includes the subscript H.

\glsxtrLatinK

A mini-rule that just covers “K” but includes the subscript K.

\glsxtrLatinI

A mini-rule that just covers “I” but includes the subscript I.

\glsxtrLatinM

A mini-rule that just covers “M” but includes the subscript M.

\glsxtrLatinN

A mini-rule that just covers “N” but includes the sub- and superscript N.

\glsxtrLatinO

A mini-rule that just covers “O” but includes the sub- and superscript O.

\glsxtrLatinP

A mini-rule that just covers “P” but includes the subscript P.

\glsxtrLatinS

A mini-rule that just covers “S” but includes the subscript S.

\glsxtrLatinT
11. *bib2gls: Managing Reference Databases*

A mini-rule that just covers “T” but includes the subscript T.

\latex{\text{\glstextrm{LatinX}}}

A mini-rule that just covers “X” but includes the subscript X.

\latex{\text{\glstextrm{LatinEszettSs}}}

A mini-rule that just covers eszett (ß) and makes “ſ” (long s followed by short s) equivalent to “ß”. (This is used in the above blocks that treat “ß” as “ss”.)

\latex{\text{\glstextrm{LatinEszettSz}}}

A mini-rule that just covers eszett (ß) and makes “ſz” (long s followed by z) equivalent to “ß”. (This is used in the above blocks that treat “ß” as “sz”.)

\latex{\text{\glstextrm{LatinEth}}}

A mini-rule for eth (Ð, ð) so you don’t need to remember the Unicode values.

\latex{\text{\glstextrm{LatinThorn}}}

A mini-rule for thorn (Þ, þ) so you don’t need to remember the Unicode values.

\latex{\text{\glstextrm{LatinAELigature}}}

A mini-rule for ae-ligature (Æ, æ) so you don’t need to remember the Unicode values.

\latex{\text{\glstextrm{LatinOELigature}}}

A mini-rule for oe-ligature (Œ, œ) so you don’t need to remember the Unicode values.

\latex{\text{\glstextrm{LatinOslash}}}
11. \textit{bib2gls}: Managing Reference Databases

A mini-rule for o-slash (Ø, ø) so you don’t need to remember the Unicode values.

\glsxtrLatinLslash

A mini-rule for l-slash (Ł, ł) so you don’t need to remember the Unicode values.

\glsxtrLatinWynn

A mini-rule for wynn (Ƿ, ƿ) so you don’t need to remember the Unicode values.

\glsxtrLatinInsularG

A mini-rule for insular-G (ǣ, ã) so you don’t need to remember the Unicode values. (Not used in any of the provided Latin rule blocks described above.)

\glsxtrLatinSchwa

A mini-rule for schwa (Ə, ə, ɘ) so you don’t need to remember the Unicode values. (Not used in any of the provided Latin rule blocks described above.)

\glsxtrLatinAA

A mini-rule for “a with ring above” (Å, å) so you don’t need to remember the Unicode values. (Not used in any of the provided Latin rule blocks described above.)

11.5.2.1.3. Math Greek

\glsxtrMathGreekIrules

A rule block for mathematical Greek (\alpha, \beta etc) and upright Greek (\upalpha, \upbeta etc, from the upgreek package) characters that includes digamma (\digamma and \Digamma) between epsilon and zeta. The upright and italic versions are gathered together into the same letter group.

\glsxtrMathGreekIIrules
11. **bib2gls: Managing Reference Databases**

As \glsxtrMathGreekIrules but doesn’t include digamma.

\begin{itemize}
  \item \glsxtrMathUpGreekIrules
    A rule block for upright Greek (\upalpha, etc, from the upgreek package) characters that includes digamma (\digamma and \Digamma) between epsilon and zeta.
  \item \glsxtrMathUpGreekIIrules
    A rule block for upright Greek (\upalpha, etc, from the upgreek package) that doesn’t include digamma.
  \item \glsxtrMathItalicGreekIrules
    A rule block for mathematical Greek (\alpha, \Alpha, etc) characters that includes digamma (\digamma and \Digamma) between epsilon and zeta. Note that even though the uppercase \Delta etc are actually rendered upright by \LaTeX, bib2gls’s interpreter treats them as italic to help keep them close to the lowercase versions.
  \item \glsxtrMathItalicGreekIIrules
    A rule block for mathematical Greek (\alpha, \Alpha, etc) characters that doesn’t include digamma.
  \item \glsxtrMathItalicUpperGreekIrules
    A rule block for uppercase mathematical Greek (\Alpha, \Beta, etc) characters that includes digamma (\Digamma) between epsilon and zeta.
  \item \glsxtrMathItalicUpperGreekIIrules
    A rule block for uppercase mathematical Greek (\Alpha, \Beta, etc) characters that doesn’t include digamma.
  \item \glsxtrMathItalicLowerGreekIrules
\end{itemize}
A rule block for lowercase mathematical Greek (\(\alpha, \beta, \text{etc}\)) characters that includes digamma (\(\digamma\)) between epsilon and zeta.

\[\text{\texttt{\textbackslash{}glsxtrMathItalicLowerGreekIIrules}}\]

A rule block for lower case mathematical Greek (\(\alpha, \beta, \text{etc}\)) characters that doesn’t include digamma.

Additionally, there are commands that just cover the uppercase and lowercase forms of a special Greek character (\(\Upalpha, \upalpha \text{etc and } \Alpha, \alpha \text{etc}\): \texttt{\textbackslash{}glsxtrUpAlpha, \textbackslash{}glsxtrMathItalicAlpha, \textbackslash{}glsxtrUpBeta, \textbackslash{}glsxtrMathItalicBeta, \textbackslash{}glsxtrUpGamma, \textbackslash{}glsxtrMathItalicGamma, \textbackslash{}glsxtrUpDelta, \textbackslash{}glsxtrMathItalicDelta, \textbackslash{}glsxtrUpEpsilon, \textbackslash{}glsxtrMathItalicEpsilon, \textbackslash{}glsxtrUpDigamma, \textbackslash{}glsxtrMathItalicDigamma, \textbackslash{}glsxtrUpZeta, \textbackslash{}glsxtrMathItalicZeta, \textbackslash{}glsxtrUpEta, \textbackslash{}glsxtrMathItalicEta, \textbackslash{}glsxtrUpTheta, \textbackslash{}glsxtrMathItalicTheta, \textbackslash{}glsxtrUpIota, \textbackslash{}glsxtrMathItalicIota, \textbackslash{}glsxtrUpKappa, \textbackslash{}glsxtrMathItalicKappa, \textbackslash{}glsxtrUpLambda, \textbackslash{}glsxtrMathItalicLambda, \textbackslash{}glsxtrUpMu, \textbackslash{}glsxtrMathItalicMu, \textbackslash{}glsxtrUpNu, \textbackslash{}glsxtrMathItalicNu, \textbackslash{}glsxtrUpXi, \textbackslash{}glsxtrMathItalicXi, \textbackslash{}glsxtrUpOmicron, \textbackslash{}glsxtrMathItalicOmicron, \textbackslash{}glsxtrUpPi, \textbackslash{}glsxtrMathItalicPi, \textbackslash{}glsxtrUpRho, \textbackslash{}glsxtrMathItalicRho, \textbackslash{}glsxtrUpSigma, \textbackslash{}glsxtrMathItalicSigma, \textbackslash{}glsxtrUpTau, \textbackslash{}glsxtrMathItalicTau, \textbackslash{}glsxtrUpUpsilon, \textbackslash{}glsxtrMathItalicUpsilon, \textbackslash{}glsxtrUpPhi, \textbackslash{}glsxtrMathItalicPhi, \textbackslash{}glsxtrUpChi, \textbackslash{}glsxtrMathItalicChi, \textbackslash{}glsxtrUpPsi, \textbackslash{}glsxtrMathItalicPsi, \textbackslash{}glsxtrUpOmega, \text{and } \texttt{\textbackslash{}glsxtrMathItalicOmega}.

Additionally, there are commands for math italic partial differential \(\partial (0x1D715)\) \texttt{\textbackslash{}glsxtrMathItalicPartial} and nabla \(\nabla (0x1D6FB)\) \texttt{\textbackslash{}glsxtrMathItalicNabla}.

### 11.5.3. Commands Used Within Resource Files

A \texttt{glstex} file will typically start with \texttt{\textbackslash{}glsnoexpandfields}, since supporting field expansion is only necessary where entry definitions may be programmatically performed within a loop or some other command. (This may be suppressed.)

If you use the \texttt{set-widest} resource option, \texttt{bib2gls} v1.8+ will write the following command in the \texttt{glstex} file:

\[\text{\texttt{\textbackslash{}glsxtrSetWidest\{}\langle{}\text{\texttt{type}}}\}{\langle{}\text{\texttt{level}}}\}\]

This sets the widest name for the given glossary type and hierarchical level. This supports both the alttree style and the styles provided by glossary-longextra, which need to know the widest name.

If \texttt{bib2gls} can’t determine the widest name (typically because the name field consists of commands that aren’t recognised by the interpreter) then \texttt{bib2gls} v1.8+ will write the
following to the glstex file:

\glsxtrSetWidestFallback\{⟨type⟩\{⟨level⟩\}

{max depth}{list} Currently the maximum hierarchical depth \(\text{max depth}\) may only be 0 or 2. This command requires commands provided by the glossaries-extra-stylemods package with the alttree style enabled. In this case, it may be simpler to just use \glssetwidest.

The glstex files may also contain instances of \glsxtrprovidestoragekey to provide new glossary entry keys or \provideignoreglossary to provide glossaries to ensure they are defined.

The \bibgls\ldots commands provided by bib2gls are defined in the glstex files with \providecommand. See the bib2gls manual for further details of those commands.

### 11.5.4. Hierarchy

The parent key identifies a parent entry (by its label), so determining if an entry has a parent can easily be achieved by testing if the parent field has been set (using \ifglshasparent or a command like \glsxtrifhasfield).

The other way around, testing if an entry has any children, is much harder. The base glossaries package provides \ifglshaschildren, which is very inefficient as it has to iterate over all entries in the entry’s glossary to determine which ones have a matching parent field.

It’s much easier with bib2gls, as there are resource options available to save this information. The save-child-count option saves the child count for each entry in the childcount internal field. This total only includes the child entries that have been selected by the resource set.

\GlsXtrIfHasNonZeroChildCount\{⟨entry-label⟩\}{⟨true⟩}{⟨false⟩} modifier: *

A shortcut that uses \GlsXtrIfFieldNonZero to test if the value supplied in the childcount field is non-zero. The starred form uses the starred form of \GlsXtrIfFieldNonZero.

### 11.5.5. Supplemental Locations

Locations in external documents can be manually added by explicitly setting the value when an entry is indexed. However, this is a bit inconvenient so bib2gls provides a way to do this automatically. Both the main document and the supplemental document need to use the record option, and the entries provided via src in the main document must have the same labels as those in the supplementary document. The supplemental document is identified by
the \texttt{supplemental-locations} resource option. See the \texttt{bib2gls} manual for further details.

\begin{verbatim}
\glsxtrdisplaysuppploc\{⟨prefix⟩\}\{⟨counter⟩\}\{⟨format⟩\}\{⟨src⟩\}\{⟨location⟩\}
\end{verbatim}

This is used by \texttt{bib2gls} version v1.7+ for supplemental locations, instead of using \texttt{\gls-xtrsupphypernumber} with the \texttt{externallocation} attribute. This command sets up the location counter and prefix (used in the formation of hyperlinks) and then uses:

\begin{verbatim}
\glsxtrmultisupplocation\{⟨src⟩\}\{⟨location⟩\}\{⟨format⟩\}
\end{verbatim}

\texttt{modifier: *}

...to format the actual location (with an external hyperlink, if supported). The \texttt{⟨format⟩} (the original location \texttt{encap}) is ignored by default.

\subsection*{11.5.6. Nameref Records}

Nameref records include the current title and hypertarget. These records are enabled with \texttt{record=nameref} and provide a more reliable way of saving the location hypertarget, which can’t be obtained with \texttt{makeindex} or \texttt{xindy}.

This option is best used with \texttt{counter=chapter} or \texttt{counter=section} if you want the title included in the location list. If the indexing counter is the default page, only the location number is shown. Similarly for \texttt{counter=equation} (or \texttt{equations=true}).

Normally locations are recorded in the aux file in the form:

\begin{verbatim}
\glsxtr@record\{⟨entry-label⟩\}\{⟨h-prefix⟩\}\{⟨counter⟩\}\{⟨encap⟩\}\{⟨location⟩\}
\end{verbatim}

This is similar to the command used with \texttt{\makenoidxglossaries} and provides the same information about the location and how to form the hypertarget as is passed to \texttt{makeindex} and \texttt{xindy}. The \texttt{record=nameref} option, which requires at least \texttt{bib2gls} v1.8, instead uses:

\begin{verbatim}
\glsxtr@record@nameref\{⟨entry-label⟩\}\{⟨h-prefix⟩\}\{⟨counter⟩\}\{⟨encap⟩\}
\{⟨location⟩\}\{⟨current title⟩\}\{⟨current anchor⟩\}\{⟨the-h-counter⟩\}
\end{verbatim}

\texttt{where} \texttt{⟨title⟩} is obtained from \texttt{\@currentlabelname} and \texttt{⟨href⟩} is obtained from \texttt{\@current-Href}. These commands require \texttt{hyperref}. If they are undefined, \texttt{⟨title⟩} and \texttt{⟨href⟩} will be left empty and \texttt{bib2gls} will treat it as a regular record.

555
Be careful with this option as \textit{href} will globally change on every instance of \texttt{\refstepcounter} but \textit{title} won’t necessarily change. It can therefore cause unexpected behaviour.

The final argument \textit{hcounter} is obtained from \texttt{\theH\texttt{counter}} which provides the partial target name associated with the location counter. With the original makeindex/xindy approach, it’s not possible to include this information in the location, so the base glossaries package attempts to derive a prefix from which the \textit{hcounter} value can be reconstituted by appending the prefix. Unfortunately, not all definitions of \texttt{\theH\texttt{counter}} are in the form \texttt{\prefix}\texttt{\the(counter)} (most notably the equation counter with chapters) so this can fail.

Since bib2gls is customized specifically for use with glossaries-extra, it’s now possible to save \textit{hcounter}, so the \texttt{recordnameref} option does this. By providing both \textit{href} and \textit{hcounter}, you can determine which target you would rather use. The default is to use \textit{hcounter}, which will take you to the place where the corresponding counter was incremented with \texttt{\refstepcounter}. However, you may choose to switch to using the \textit{href} target, which will take you to the nearest target before the indexing took place.

With bib2gls v1.8+, normal locations are displayed using:

\begin{verbatim}
\glssnoidxdisplayloc{\texttt{\prefix}}{\texttt{\counter}}{\texttt{\format}}{\texttt{\location}}
\end{verbatim}

This is provided by the base glossaries package and is simply defined to do:

\begin{verbatim}
\setentrycounter[\texttt{\prefix}]{\texttt{\counter}}{\texttt{\format}}{\texttt{\location}}
\end{verbatim}

Earlier versions of bib2gls only used this in the \texttt{loclist} field and explicitly used \texttt{\setentrycounter} in the \texttt{location} field followed by \texttt{\\texttt{\format}}{\texttt{\location}}, which follows the code that’s created with the default makeindex setting. The \texttt{\setentrycounter} command sets up the prefix needed for \texttt{\glshypernumber} to reform the target name from the given location.

The locations identified by \texttt{\glsxtr@record@nameref} are written by bib2gls to the location list using:

\begin{verbatim}
\glsxtrdisplaylocnameref{\texttt{\prefix}}{\texttt{\counter}}{\texttt{\format}}{\texttt{\location}}{\texttt{\title}}{\texttt{\href}}{\texttt{\hcounter}}{\texttt{\file}}
\end{verbatim}

With normal internal locations, \texttt{\file} will always be empty. With supplemental locations, \texttt{\file} will be the external file reference. If \texttt{hyperref} hasn’t been loaded, this command behaves like \texttt{\glssnoidxdisplayloc}, so it will simply encapsulate the location with the given formatting command.

If \texttt{hyperref} has been loaded, then \texttt{\glsxtrdisplaylocnameref} will try to work out the
appropriate hyperlink anchor and text. The \textit{prefix} argument is redundant. It first defines the following commands:

\begin{verbatim}
\glsxtrrecentanchor
\end{verbatim}

This expands to the \textit{href} argument, which corresponds to the \texttt{\@currentHref} value at the time the location was recorded. If this is used as the anchor, the link will go to the most recent anchor before the record was created. This is more likely to match the given title, but won’t necessarily match the corresponding counter.

For example, if the record was created with \texttt{counter=section} but occurred in a table caption, then \texttt{\glsxtrrecentanchor} and the title will correspond to the table caption. If you have defined \texttt{\glsxtrsectionlocfmt} to show the section number (see below), then this may cause some confusion if clicking on the section number leads to a table caption. However, it will lead to the closest location to where the record was created, which may be preferred.

\begin{verbatim}
\glsxtrlocationanchor
\end{verbatim}

This expands to the anchor that corresponds to the record’s location counter, which is constructed from the \textit{counter} and \textit{hcounter} arguments.

In the above example with the section counter, \textit{hcounter} will be the value of \texttt{\theHsection} when the record was created, so the constructed anchor will be \texttt{section}. \textit{hcounter} which corresponds to the anchor at the start of the section. However, if you haven’t defined \texttt{\glsxtrsectionlocfmt}, then the title will correspond to the table caption, which may be a little confusing.

The actual anchor used is obtained from the expansion of:

\begin{verbatim}
\glsxtractualanchor
\end{verbatim}

This is initialised to the value of \texttt{\glsxtrlocationanchor} but may be changed by:

\begin{verbatim}
\glsxstrsetactualanchor{\textit{counter}}
\end{verbatim}

The argument is the counter name, which may be used to set choose an alternative anchor depending on the counter. This command does nothing by default, which means that \texttt{\glsxtrlocationanchor} will be used.

For example, to switch to \texttt{\glsxtrrecentanchor} if the counter is page:
If you are using `indexcounter` then the recent anchor will be the one created by the `wrglossary` increment just before the indexing occurs. This means that with `counter = wrglossary`, the location anchor and recent anchor will be the same. With other counters, the recent anchor will be the closest anchor to the place where indexing occurred.

To allow for different formatting according to the counter name, \texttt{\glsxtrdisplaylocnameref} first checks for the existence of:

\begin{verbatim}
\glsxtr{counter}locfmt\{\langle location\rangle\}\{\langle title\rangle\}
\end{verbatim}

If this command is defined, the location will be displayed using:

\begin{verbatim}
\glsxtrnamereflink\{\langle format\rangle\}\glsxtr{counter}locfmt\{\langle location\rangle\}\{\langle title\rangle\}
\{\langle href\rangle\}\{\langle file\rangle\}
\end{verbatim}

Note the above warning about the possible mismatch of the title with the anchor. For example, if the following is defined for the section counter:

\begin{verbatim}
\newcommand{\glsxtrsectionlocfmt}[2]{\S#1 (#2)}
\end{verbatim}

then this could lead to the section number followed by the table caption. A more compact form that omits the title is better:

\begin{verbatim}
\newcommand{\glsxtrsectionlocfmt}[2]{\S#1}
\end{verbatim}

The following location formats are predefined. The equation counter:

\begin{verbatim}
\glsxtrequationlocfmt\{\langle location\rangle\}\{\langle title\rangle\}
\end{verbatim}

This simply expands to \langle \langle location\rangle \rangle since equations typically don’t have a title, but are usu-
ally enclosed in parentheses.

\glsxtrwrglossarylocfmt\{location\}\{title\}

This is used when the \texttt{indexcounter} option creates records with the \texttt{wrglossary} counter. This ensures that the page name is shown rather than the value of the \texttt{wrglossary} counter.

If the corresponding \texttt{\glsxtr\{counter\}\locfmt} hasn’t been defined, \glsxtr\texttt{displaylocnameref} will do one of the following:

- if \langle title \rangle is empty or the counter is page, \glsxtr\texttt{nameref} with the title set to the location;
- otherwise it will do:

\glsxtr\texttt{titlednameref}\{format\}\{location\}\{title\}\{file\}

which uses \glsxtr\texttt{nameref} with the given \langle title \rangle and \glsxtr\texttt{recentanchor} as the anchor. This has a better chance of matching the title with the anchor, but it’s not guaranteed as some anchors are created without a title. This is defined as:

\texttt{\newcommand{\glsxtr\texttt{titlednameref}}[4]{\%}\
\glsxtr\texttt{nameref}\{#1\}\{#2\}\{\glsxtr\texttt{recentanchor}\}\{#4\}\%}\
}

This shows the formatted title with the recent anchor. The location isn’t shown. If you would prefer to just show the location and use the anchor corresponding to the location counter:

\texttt{\renewcommand{\glsxtr\texttt{titlednameref}}[4]{\%}\
\glsxtr\texttt{nameref}\{#1\}\{#2\}\{\glsxtr\texttt{locationanchor}\}\{#3\}\%}\
}

If \langle file \rangle is empty an internal link is created with:

\glsxtr\texttt{fmtinternalnameref}\{target\}\{format\}\{file\}

otherwise an external link is created with:

\glsxtr\texttt{fmtexternalnameref}\{target\}\{format\}\{title\}\{file\}
The <file> argument is set by bib2gls for supplemental locations (obtained from \supplementallocations).

The link is encapsulated with the text-block command whose name is given by <format>, but \glshypernumber is first locally redefined to \firstofone to prevent a conflict with the usual location hyperlink formation. This means that if the <format> is \hyperbf then it will simply behave like \textbf.

The following command is provided but not used by default:

```
glsxtrnameloclink{<prefix>}{<counter>}{<format>}{<location>}{<text>}{<file>}
```

This creates a hyperlink to the target in the given external file obtained from the <prefix>, <counter> and <location> but uses <text> as the hyperlink text. As with regular indexing, this will fail if the target name can’t be formed by prefixing the location value.

For compactness, bib2gls merges normal records if the <prefix>, <counter> and <location> all match. (An order of precedence can be provided for format conflicts.) With nameref records, you can use the --merge-nameref-on switch provided by bib2gls v1.8+ to determine how to merge nameref records. This switch must be followed by one of the following keywords: hcounter (merge on <hcounter>, default) href (merge on <href>), title (merge on <title>) and location (merge on <location>, as regular records). In all cases, the <counter> must also match.

### 11.5.7. Dual Entry Commands

Dual entries can be defined with entry types like @dualentry or @dualindexabbreviation. A single entry definition within the bib file creates two dependent entries that may be referenced within the document and assigned to different glossaries. The default selection= {recorded and deps} will ensure that dependent entries are selected, even if they don’t have any records. For example, the following bib entry:

```
@dualindexsymbol{L,
    name={Lagrangian function},
    symbol={\ensuremath{L}},
    description={a function of generalized co-ordinates}
}
```

is essentially like:

```
@index{L,
    name={Lagrangian function},
    symbol={\ensuremath{L}},
}
```
but additionally the two entries (the primary \( L \) and the dual \( \text{dual.L} \)) are dependent. The following document only references the primary entry, but the dependency ensures that that the dual is also selected:

```
\usepackage[record,symbols,nostyles,stylemods={tree,bookindex}]{glossaries-extra}
\GlsXtrLoadResources[src=entries,dual-type=symbols]
\glsdefpostlink{index}{\glsxtrpostlinkAddDescOnFirstUse}
\glsdefpostname{index}{\ifglshassymbol{\glscurrententrylabel}{ (\glsentrysymbol{\glscurrententrylabel})}{}\}
\begin{document}
Primary: \gls{L}.
\printunsrtglossary[type=symbols,style=tree]
\printunsrtglossary[type=Index,style=bookindex]
\end{document}
```

The dependency ensures that the dual entry \( \text{dual.L} \) is selected, and the \texttt{dual-type} setting adds the dual entry to the \texttt{symbols} glossary (which was defined with the \texttt{symbols} package option).

It may be useful to have a hyperlink from the entry in one glossary to its dependent in the other glossary. This can be achieved by instructing \texttt{bib2gls} to save the label of the entry’s opposite using \texttt{dual-field}. The value of this resource option indicates the field in which to save the label. If omitted, \texttt{dual} is assumed.

The glossary style can then be adapted to check if the field has been set and, if so, to create a hyperlink. The following command is provided to assist with this:

```
\GlsXtrDualBackLink{⟨text⟩}{⟨entry-label⟩}
```

This is defined as:
This obtains the field from:

\GlsXtrDualField

This expands to dual, which is the default for dual-field if no value is supplied.

The above example can be adapted as follows:

\usepackage[colorlinks]{hyperref}
\usepackage[record,symbols,nostyles,stylemods={tree,bookindex}]{glossaries-extra}
\GlsXtrLoadResources[src=entries, dual-type=symbols,dual-field]
\glsdefpostlink{index}{\glsxtrpostlinkAddSymbolOnFirstUse}
\glsdefpostname{index}{\ifglshassymbol{\glscurrententrylabel}{\GlsXtrDualBackLink{\glssentrysymbol{\glscurrententrylabel}}}{\glscurrententrylabel}}}{}
\begin{document}
Primary: \gls{L}.
\printunsrtglossary[type=symbols,style=tree]
\printunsrtglossary[title=Index,style=bookindex]
\end{document}

Example 158: Using bib2gls: dual backlinks

Primary: Lagrangian function \(L\).

Symbols

\(L\) (Lagrangian function) a function of generalized co-ordinates

Index

Lagrangian function \(L\), 1
This checks if the symbol key has been set in both the category post-link hook and the category post-name hook to append the symbol in parentheses. In addition, the category post-name hook adds a link to the corresponding dual entry.

If you prefer instead to have the backlink on the name in both glossaries, then this can more easily be achieved with a resource option such as dual-backlink.

The commands described below were designed for use with bib2glsl’s dual entries, but may also be used in other contexts where a label may potentially have a number of possible prefixes.

It’s possible to use commands like \glsxtrnewgls to create \gls-like commands that automatically insert a label prefix (such as dual. for dual entries). The commands described in this section provide a similar set of \gls-like commands that iterate over a set of possible prefixes until a match is found.

Each possible prefix (which may be empty) is identified by:

\glsxtraddlabelprefix{⟨label-prefix⟩}

{prefix} These should be listed in order of precedence.

You can prepend a prefix to the list using:

\glsxtrprependlabelprefix{⟨label-prefix⟩}

which gives it the highest order of precedence.

The list of known prefixes can be (locally) cleared with:

\glsxtrclearlabelprefixes

You can test if a prefix is already in the list with:

\glsxtrifinlabelprefixlist{⟨label-prefix⟩}{⟨true⟩}{⟨false⟩}

This does ⟨true⟩ if the given prefix has been added to the list. No expansion is performed on ⟨label-prefix⟩.

In the event that there’s no match for any of the prefixes (which will occur on the first \LaTeX run before the \texttt{glistex} file has been created), the fallback is determined by the conditional:

\ifGlsXtrPrefixLabelFallbackLast ⟨true⟩\else ⟨false⟩\fi
If true, this will fallback on the last prefix, otherwise it will fallback on the first. This conditional can be set to true with:

\GlsXtrPrefixLabelFallbackLasttrue

and to false with:

\GlsXtrPrefixLabelFallbackLastfalse

The default is true.

As from v1.49, all possible matches will be recorded using a special syntax if none are found. This ensures there will be at least one match on the first run. However, note that this requires bib2gls v3.0+.

In general it’s best to avoid adding multiple instances of the same prefix, so you can check with this command before adding a prefix to the list. However, it can be useful to repeat a prefix at the start or end of the list so that it can be used as a fallback for entries that haven’t yet been defined.

With the list of possible prefixes set up (including an empty prefix if necessary), you can use:

\dgls[⟨options⟩]{⟨entry-label⟩}[[⟨insert⟩]] modifiers: * + ⟨alt-mod⟩

which behaves like

\gls[⟨options⟩]{⟨prefix⟩⟨entry-label⟩}[[⟨insert⟩]]

where ⟨prefix⟩ is the first prefix in the list such that ⟨prefix⟩⟨label⟩ matches a defined entry. This requires bib2gls v3.0+ to work properly on the first \TeX run (when no entries are defined).

There are also analogous commands for the plural and case-changing versions:

\dglspl[⟨options⟩]{⟨entry-label⟩}[[⟨insert⟩]] modifiers: * + ⟨alt-mod⟩

\dglspl[⟨options⟩]{⟨entry-label⟩}[[⟨insert⟩]] modifiers: * + ⟨alt-mod⟩
11. bib2gls: Managing Reference Databases

\begin{verbatim}
(uses \glsp),
\texttt{\textbackslash dGls}[\langle options\rangle] \{ \langle entry-label\rangle \}[\langle insert\rangle] \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \Glsp),

\begin{verbatim}
\texttt{\textbackslash dGlp}\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \GLp),

\begin{verbatim}
\texttt{\textbackslash dGLS}\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \GLSp),

\begin{verbatim}
\texttt{\textbackslash dGLSp}\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \GLSp),

\begin{verbatim}
\texttt{\textbackslash dglslink}[\langle options\rangle] \{ \langle entry-label\rangle \}\{\langle text\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \glslink),

\begin{verbatim}
\texttt{\textbackslash dGlslink}[\langle options\rangle] \{ \langle entry-label\rangle \}\{\langle text\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \Glslink),

\begin{verbatim}
\texttt{\textbackslash dglslisp}\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle text\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \glslisp) and

\begin{verbatim}
\texttt{\textbackslash dGlsdisp}[\langle options\rangle] \{ \langle entry-label\rangle \}\{\langle text\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

(uses \Glsp).
If you want to use a specific field you can instead use:

\dglSfield\[\langle options\rangle\]{\langle entry-label\rangle}{\langle field-label\rangle}{\langle text\rangle} 
\langle alt-mod\rangle 

modifiers: * +

This will find the first match that has the given field set (that is, the field value has been defined and is not empty or \relax). The field should be identified by its internal field label. There are also case-changing versions:

\dGlsfield\[\langle options\rangle\]{\langle entry-label\rangle}{\langle field-label\rangle}{\langle text\rangle} 
\langle alt-mod\rangle 

\dGLSfield\[\langle options\rangle\]{\langle entry-label\rangle}{\langle field-label\rangle}{\langle text\rangle} 
\langle alt-mod\rangle 

which applies a sentence case change and

which applies all caps. Note that at least one of the potential labels must have the given field set in order for the reference to be correctly resolved. For example:

\dglSfield{pi}{symbol}

If you find this a bit long-winded to type and you want to provide a shorter command that recognises the modifiers, then you can use:

\newdglSfield\[\langle default-options\rangle\]{\langle field\rangle}{\langle cs\rangle}

For example:

\newdglSfield{symbol}{\sym}
\sym{pi}

If you also want sentence case and all caps versions use:

\newdglSfieldlike\[\langle default-options\rangle\]{\langle field\rangle}{\langle cs\rangle}{\langle Cs\rangle}{\langle CS\rangle}
where \(\langle cs\rangle\) is the command without a case-change (which will use \(\texttt{dglsfield}\)), \(\langle Cs\rangle\) is the sentence case command (which will use \(\texttt{dGlsfield}\)) and \(\langle CS\rangle\) is the all caps command (which will use \(\texttt{dGLSfield}\)). This will also use \texttt{glsmfuaddmap} and \texttt{glsmfublocker} to establish the sentence case mapping from \(\langle cs\rangle\) to \(\langle Cs\rangle\) and block the case change for \(\langle CS\rangle\).

No expansion is performed on \(\langle\text{default-options}\rangle\) or \(\langle\text{field}\rangle\) while the new commands are being defined.

This family of \texttt{dglslfield} commands all define:

\begin{verbatim}
\texttt{\textbackslash dglsfieldcurrentfieldlabel}
\end{verbatim}

This expands to the given field label.
If the requested field isn’t set (according to \texttt{\textbackslash ifcsvoid}) then the fallback field will be tried instead.

\begin{verbatim}
\texttt{\textbackslash dglsfieldfallbackfieldlabel initial: text}
\end{verbatim}

This expands to the fallback field, which defaults to the \texttt{text} field.
If you need to know whether the requested field or the fallback field was used, the following will be set to the actual field used.

\begin{verbatim}
\texttt{\textbackslash dglsfieldactualfieldlabel}
\end{verbatim}

For example, suppose the file \texttt{entries.bib} contains:

```latex
@index{duck}
@dualindexabbreviation{svm, 
  short={SVM},
  long={support vector machine}
}
@dualindexsymbol{pi, 
  symbol={\ensuremath{\pi}},
  description=ratio of a circle's circumference to its diameter
}
```

and suppose the document code is:
\documentclass{article}
\usepackage{hyperref}
\usepackage[record,abbreviations,symbols]{glossaries-extra}

\newcommand{\bibglsnewdualindexsymbolsecondary}{5}{% \longnewglossaryentry*{#1}{name={#3},category={symbol}, type={symbols},symbol={#4},#2}{#5}% }
\GlsXtrLoadResources[src=entries]

\begin{document}
First use: \gls{duck}, \gls{svm}, \gls{pi}.
Next use: \gls{duck}, \gls{svm}, \gls{pi}.
\printunsrtglossaries
\end{document}

This uses the default empty primary prefix and dual. For the dual prefix, so \gls{svm} is referencing the primary entry, which is (essentially) an @index type not an abbreviation. It therefore doesn’t follow the abbreviation style, and it also hyperlinks to the index not to the list of abbreviations. Similarly for \gls{pi}, which references the primary @index entry rather than the symbol.

What’s really needed is:

\gls{duck}, \gls{dual.svm}, \gls{dual.pi}.

or use label-prefix and dual-prefix to set the label prefixes:

\GlsXtrLoadResources[src=entries, label-prefix={idx.}, dual-prefix={}]

then only the entries without a dual need a prefix:

\gls{idx.duck}, \gls{svm}, \gls{pi}.

Using \glsxtrnewgls or \glsxtrnewglslike (see §5.7) to define the custom \idx:
the entry references can be simplified to:

\idx{duck}, \gs{svm}, \gs{pi}.

but this requires remembering which terms have duals. An alternative is to use \dgs instead:

\GlsXtrLoadResources[src=entries, combine-dual-locations=primary]
\glsxtraddlabelprefix{dual.}
\glsxtraddlabelprefix{}
\begin{document}
First use: \dgs{duck}, \dgs{svm}, \dgs{pi}.
Next use: \dgs{duck}, \dgs{svm}, \dgs{pi}.
\printunsrtglossaries
\end{document}
Example 159: Using \texttt{bib2gls}: dual entry label prefixes

First use: duck, support vector machine (SVM), \( \pi \). Next use: duck, SVM, \( \pi \).

**Glossary**

\textit{duck} 1

\textit{pi} ratio of a circle’s circumference to its diameter 1

\textit{SVM} 1

**Symbols**

\( \pi \) ratio of a circle’s circumference to its diameter

**Abbreviations**

\textit{SVM} support vector machine

On the first \texttt{\LaTeX} call (when the \texttt{glstex} file doesn’t exist), neither \textit{dual.svm} nor \textit{svm} exists, so \texttt{\dgls} uses the last prefix (which is empty in this case). This means that on the first \texttt{\LaTeX} run, \texttt{\dgls\{svm\}} behaves like \texttt{\gls\{svm\}}, which adds a record for the primary \textit{svm} entry. The default primary-dual dependency means that this will cause both the primary (\textit{svm}) and dual (\textit{dual.svm}) entry to be selected. The location will be added to the primary entry’s location list, unless overridden by resource options, such as \texttt{combine-dual-locations}.

Once \texttt{bib2gls} has been run and the \texttt{glstex} file exists, then \textit{dual.svm} exists. So \texttt{\dgls \{svm\}} will again first try \texttt{\dgls\{dual.svm\}} (as \texttt{dual.svm} is the first in the list of label prefixes). That now exists, so \texttt{\dgls\{svm\}} now behaves like \texttt{\dgls\{dual.svm\}}, which follows the abbreviation style and hyperlinks to the list of abbreviations. Similarly for the index-symbol combination \textit{dual.pi} and \textit{pi}.

In the case of \texttt{\dgls\{duck\}}, which doesn’t have a dual, the label \textit{dual.\texttt{duck}} never exists, so that’s never selected. However, when there’s no match, such as when the \texttt{glstex} file doesn’t exist, the duck entry will be recorded with both the \texttt{dual.} prefix and the empty prefix. This allows \texttt{bib2gls} to test which prefix-label combination matches.
If you haven't used `combine-dual-locations` an extra `bib2gls+pdflatex` run may be required to correct the location lists.

If you change the label prefixes, remember to update the corresponding `\glsxtraddlabelprefix{⟨label-prefix⟩}`. If no prefixes have been added to the list (or if the list is cleared), just an empty prefix is assumed.

Note that `bib2gls` v1.8+ provides hooks that identify the label prefixes in the `glstex` file:

\begin{verbatim}
\bibglstertiaryprefixlabel{⟨label-prefix⟩}
\bibglsdualprefixlabel{⟨label-prefix⟩}
\bibglsprimaryprefixlabel{⟨label-prefix⟩}
\end{verbatim}

These do nothing by default, but they can be defined before the resource file is loaded to set up the prefix list. For example:

\begin{verbatim}
\newcommand{\bibglsprimaryprefixlabel}{[1]{% \glsxtraddlabelprefix{#1}}
\newcommand{\bibglsdualprefixlabel}{[1]{% \glsxtrprependlabelprefix{#1}}
\GlsXtrLoadResources[src=entries]
\end{verbatim}

Remember that this will only have an effect once the `glstex` file has been created. The prefix list will be empty on the first run (which is treated as a single empty prefix). If this isn’t a suitable fallback, it may be necessary to add one after all the resource commands:

\begin{verbatim}
\newcommand{\bibglsprimaryprefixlabel}{[1]{% \glsxtraddlabelprefix{#1}}
\newcommand{\bibglsdualprefixlabel}{[1]{% \glsxtrprependlabelprefix{#1}}
\GlsXtrLoadResources[src=entries,label-prefix={idx.}]
\glsxtraddlabelprefix{idx.}
\end{verbatim}

Although this rather defeats the purpose of using the hooks as you still have to keep track of the fallback prefix.

### 11.5.8. Supplementary Commands

\begin{verbatim}
\glsdisplaynumberlist{⟨entry-label⟩}
\end{verbatim}
This command is provided by the base glossaries package, but it requires hooking into the location list formatting within the glossary to obtain the list. This information is more easily available with bib2gls, which stores the formatted location list in the \texttt{location} field, so glossaries-extra-bib2gls redefines this command to use that field value. Likewise for:

\begin{verbatim}
\glsentrynumberlist\{⟨entry-label⟩\}
\end{verbatim}

which is redefined to simply do:

\begin{verbatim}
\glsxtrusefield\{#1\}\{location\}
\end{verbatim}

\begin{verbatim}
\IfTeXParserLib\{⟨TEX parser lib code⟩\}\{⟨\TeXX code⟩\}
\end{verbatim}

This command is defined by glossaries-extra-bib2gls to expand to \TeXX code, but is defined by the \TeX parser library used by bib2gls’s to expand to \TeX parser lib code. May be used in the @preamble or in field values to provide bib2gls with alternative content.

\begin{verbatim}
\glsxtrprovidecommand\{⟨cs⟩\}[⟨n⟩][⟨default⟩]\{⟨definition⟩\} modifier: *
\end{verbatim}

This command is intended for use in @preamble. It’s simply defined to \providecommand in glossaries-extra-bib2gls but bib2gls’s interpreter treats it as \renewcommand. This means that you can override bib2gls’s internal definition of a command without overriding the command definition in the document (if it’s already defined before the resource file is input). For example:

\begin{verbatim}
@preamble{"\glsxtrprovidecommand\{\int\}\{integral\}"}
\end{verbatim}

This will force bib2gls to treat \texttt{\int} as the word “integral” to assist sorting but if this preamble code is written to the \texttt{glstex} file (as it is by default) then it won’t override the current definition (provided by the kernel or redefined by a package).

The helper commands in the resource files are defined using \providecommand. For many of them, if you want to provide an alternative definition then you need to define the command before the resource file is loaded. There are a few that may be redefined afterwards but if you use \renewcommand then you will get an error on the first \TeX run when the \texttt{glstex} file doesn’t exist. In this case, you may prefer to use:

\begin{verbatim}
\glsrenewcommand\{⟨cs⟩\}[⟨n⟩][⟨default⟩]\{⟨definition⟩\} modifier: *
\end{verbatim}
This behaves like \renewcommand but only generates a warning rather than an error if the command isn’t already defined so it won’t interrupt the document build.

\GlsXtrIndexCounterLink{(text)}{(entry-label)}

For use with the indexcounter package option and the save-index-counter resource option. This command creates a hyperlink (if supported) to the target obtained from the indexcounter field (which will be set in the glstex file, if applicable).

If then indexcounter field is set for the entry given by (entry-label), this command does \hyperref[wrglossary.(value)]{(text)}, where (value) is the value of the indexcounter field. If the field isn’t set or if \hyperref hasn’t been defined, this just does (text). See the bib2gls manual (v1.4+) for further details.

The glossaries-extra-bib2gls package also provides definitions of the missing mathematical Greek commands: \Alpha, \Beta, \Epsilon, \Zeta, \Iota, \Kappa, \Mu, \Nu, \Omicron, \Rho, \Tau, \Chi, \Digamma, \omicron. These are all defined with \providecommand, so they won’t override any definitions provided by any package loaded before glossaries-extra. Since bib2gls’s interpreter recognises these commands, using them instead of explicitly using the Latin characters with the same shape helps to keep the Greek symbols together when sorting. Similarly, if upgreek has been loaded, the missing upright Greek commands are also provided: \Ualpha, \Ubeta, \Uepsilon, \Uzeta, \Ueta, \Uiota, \Ukappa, \Umu, \Uomicron, \Urho, \Utau, \Uchi, \upomicron.
12. Auto-Indexing

It’s possible that you may also want a normal index as well as the glossary, and you may want entries to automatically be added to the index (as in this document). There are two attributes that govern this: \texttt{indexname} and \texttt{dualindex}.

The auto-indexing is designed for \texttt{makeindex} syntax. If you’ve used the \texttt{xindy} package option, the automatic escaping of \texttt{xindy} special characters in the \texttt{sort} field may result in an incorrect sort value for the \texttt{index} command used by the auto-indexing. Note also that \texttt{texindy} has a fixed set of special characters (corresponding to \texttt{makeindex}’s defaults) that can’t be customized. You may want to consider using \texttt{bib2gls} and its dual entries as an alternative approach.

The \texttt{\glsxtrpostnamehook} macro, used at the end of \texttt{\glossentryname} and \texttt{\Glossentryname}, checks the \texttt{indexname} attribute for the category associated with that entry. Since \texttt{\glossentryname} is used in the default glossary styles, this makes a convenient way of automatically indexing each entry name at its location in the glossary without fiddling around with the value of the \texttt{name} key.

The internal macro used by the \texttt{glossaries} package to write the information to the external glossary file is modified to check for the \texttt{dualindex} attribute.

In both cases, the indexing is done through:

\begin{verbatim}
\glsxtrdoautoindexname{(entry-label)}{(attribute)}
\end{verbatim}

This uses the standard \texttt{\index} command with the sort value taken from the entry’s \texttt{sort} key and the actual value set to \texttt{\glossentryname{(entry-label)}}. There are user-level commands available to change the sort and actual value used by the automated index.

The actual value is given by:

\begin{verbatim}
\glsxtrautoindexentry{(entry-label)}
\end{verbatim}

where \texttt{(entry-label)} is the entry’s label. The default definition is:

\begin{verbatim}
\newcommand*{\glsxtrautoindexentry}[1]{\string\glossentryname{#1}}
\end{verbatim}
12. Auto-Indexing

Note the use of \string to prevent \glsentryname from being expanded as it’s written to the index file.

The sort value is assigned using:

\glsxtrautoindexassignsort{⟨entry-label⟩}

where ⟨entry-label⟩ is the entry label and ⟨cs⟩ is the command which needs to be set to the sort value. The default definition is:

\newcommand*{\glsxtrautoindexassignsort}[2]{%  
  \glsletentryfield{#1}{#2}{sort}  
}%

After this macro is called, ⟨cs⟩ is then processed to escape any of makeindex’s special characters. Note that this escaping is only performed on the sort not on the actual value. The escaping of the sort value is performed by

\glsxtrautoindexesc

You can redefine this to do nothing if you want to omit the escaping. You may want to consider providing another field to obtain the appropriate sort value if the one provided in the sort field isn’t suitable (because it may already have had special characters escaped or it may be a numeric value in the case of sort by use or definition).

The command used to perform the actual indexing is:

\glsxtrautoindex

This just does \index{⟨text⟩} by default.

The entry’s parent field isn’t referenced in this automated indexing.

For example, to index the value of the first key, instead of the name key:

\renewcommand*{\glsxtrautoindexentry}[1]{\string\glsentryfirst{#1}}

and if the sort value also needs to be set to the long field, if present, otherwise the sort field:
If the value of the attribute is “true”, no encap will be added, otherwise the encap will be the attribute value. For example:

\glssetcategoryattribute{general}{indexname}{textbf}

will set the encap to \textbf{which will display the relevant page number in bold whereas

\glssetcategoryattribute{general}{dualindex}{true}

won’t apply any formatting to the page number in the index.

The location used in the index will always be the page number not the counter used in the glossary. (Unless some other loaded package has modified the definition of \index to use some thing else.)

By default the \textit{format} key won’t be used with the \texttt{dualindex} attribute. You can allow the \textit{format} key to override the attribute value by using the preamble-only command:

\GlsXtrEnableIndexFormatOverride

If you use this command and hyperref has been loaded, then the theindex environment will be modified to redefine \glshypernumber to allow formats that use that command.

The \texttt{dualindex} attribute will still be used on subsequent use even if the \texttt{indexonly-first} attribute (or \texttt{indexonlyfirst} package option) is set. However, the \texttt{dualindex} attribute will honour the \texttt{noindex} key.

The \texttt{glsxtrdoadutoindexname} command will attempt to escape any of makeindex’s special characters, but there may be special cases where it fails, so take care. This assumes the default makeindex actual, level, quote and encap values (unless any of the commands \texttt{\actualchar}, \texttt{\levelchar}, \texttt{\quotechar} or \texttt{\encapchar} have been defined before glossaries-extra is loaded).
If this isn’t the case, you can use the following preamble-only commands to set the correct characters.

\GlsXtrSetActualChar{⟨character⟩}

Set the actual character to ⟨char⟩.

\GlsXtrSetLevelChar{⟨character⟩}

Set the level character to ⟨char⟩.

\GlsXtrSetEscChar{⟨character⟩}

Set the escape (quote) character to ⟨char⟩.

\GlsXtrSetEncapChar{⟨character⟩}

Set the encap character to ⟨char⟩.

Be very careful of possible shifting category codes!
13. On-the-Fly Document Definitions

The commands described here may superficially look like \texttt{⟨word⟩\index{⟨word⟩}}, but they behave rather differently. If you want to use \texttt{\index} then just use \texttt{\index}.

The base \texttt{glossaries} package advises against defining entries in the document environment. As mentioned in §2.4, this ability is disabled by default with \texttt{glossaries-extra} but can be enabled using the \texttt{docdef} package options.

Although this can be problematic, the \texttt{glossaries-extra} package provides a way of defining and using entries within the document environment without the tricks used with the \texttt{docdef} option. \textit{There are limitations with this approach, so take care with it}. This function is disabled by default, but can be enabled using the preamble-only command:

\begin{verbatim}
\GlsXtrEnableOnTheFly
\end{verbatim}

When used, this defines the commands described below. The starred version was provided to help support UTF-8 with \texttt{inputenc}. Recent versions of the \LaTeX kernel now provide better support. Both \texttt{glossaries} and \texttt{glossaries-extra} have been updated to help integrate the new kernel features, so the unstarrred version may also work with UTF-8.

If you use the starred version of \texttt{\GlsXtrEnableOnTheFly} don’t use any commands in the \texttt{(label)}, even if they expand to just text.

\begin{verbatim}
\glsxtr[(gls-options)][(dfn-options)]{(entry-label)}
\end{verbatim}

If an entry with the label \texttt{(entry-label)} has already been defined, this just does \texttt{\gls[(gls-options)]{(entry-label)}}. If \texttt{(entry-label)} hasn’t been defined, this will define the entry using:

\begin{verbatim}
\newglossaryentry{(entry-label)}{name={entry-label},
category={\glsxtrcat},
description={\nopostdesc},
(df\texttt{\texttt{n}-options})}
\end{verbatim}
The \langle label \rangle must contain any non-expandable commands, such as formatting commands or problematic characters. If the term requires any of these, they must be omitted from the \langle entry-label \rangle and placed in the name key within the optional argument \langle dfn-options \rangle.

The second optional argument \langle dfn-options \rangle should be empty if the entry has already been defined, since it’s too late for them. If it’s not empty, a warning will be generated with:

\GlsXtrWarning{\langle options \rangle}{\langle entry \rangle}

For example, this warning will be generated on the second instance of \glsxtr below:

\glsxtr[]\[plural=geese\]{goose}
\% ... later
\glsxtr[]\[plural=geese\]{goose}

If you are considering doing something like:

\newcommand*{\goose}{\glsxtr[]\[plural={geese}\]{goose}}
\renewcommand*{\GlsXtrWarning}[2]{}
\% ... later
\goose\ some more text here

then don’t bother. It’s simpler and less problematic to just define the entries in the preamble with \newglossaryentry and then use \gls in the document.

There are also plural and case-changing alternatives to \glsxtr.

\glsxtrpl{\langle gls-options \rangle}{\langle dfn-options \rangle}{\langle entry-label \rangle}

This is like \glsxtr but uses \glspl instead of \gls.

\Glsxtr{\langle gls-options \rangle}{\langle dfn-options \rangle}{\langle entry-label \rangle}

This is like \glsxtr but uses \Gls instead of \gls.

\Glsxtrpl{\langle gls-options \rangle}{\langle dfn-options \rangle}{\langle entry-label \rangle}
This is like \texttt{\textglsxtr} but uses \texttt{\textglsxtrpl} instead of \texttt{\textgls}.

The category is set to:

\begin{verbatim}
\textglsxtrcat initial: general
\end{verbatim}

This should simply expand to the required category label. The default definition is \texttt{general}.

The commands \texttt{\textglsxtr}, \texttt{\textglsxtrpl}, \texttt{\textglsxtr} and \texttt{\textglsxtrpl} can’t be used after the glossaries have been displayed (through \texttt{\textprintglossary} etc). It’s best not to mix these commands with the standard glossary commands, such as \texttt{\textgls} or there may be unexpected results.
14. Supplementary Files

The glossaries-extra package comes with some additional files. Those listed in §14.1 provide dummy entries for testing various styles. They should be placed on \TeX’s path.

There are also some sample files listed in §14.2. These should be located in the package documentation directory.

14.1. Dummy Files for Testing

The base glossaries package provides files with dummy entries for testing. The glossaries-extra package provides an additional file with entries.

- \texttt{example-glossaries-xr.tex}
  This file contains entries that have the see, seealso or alias keys set.
  There are also bib files corresponding to all the available \texttt{tex} files for use with \texttt{bib2gls}.

- \texttt{example-glossaries-acronym.bib}
  Corresponds to \texttt{example-glossaries-acronym.tex}

- \texttt{example-glossaries-acronym-desc.bib}
  Corresponds to \texttt{example-glossaries-acronym-desc.tex}

- \texttt{example-glossaries-acronyms-lang.bib}
  Corresponds to \texttt{example-glossaries-acronyms-lang.tex}

- \texttt{example-glossaries-brief.bib}
  Corresponds to \texttt{example-glossaries-brief.tex}

- \texttt{example-glossaries-childmultipar.bib}
  Corresponds to \texttt{example-glossaries-childmultipar.tex}

- \texttt{example-glossaries-childnoname.bib}
  Corresponds to \texttt{example-glossaries-childnoname.tex}

- \texttt{example-glossaries-cite.bib}
  Corresponds to \texttt{example-glossaries-cite.tex}

- \texttt{example-glossaries-images.bib}
  Corresponds to \texttt{example-glossaries-images.tex}

- \texttt{example-glossaries-long.bib}
14. Supplementary Files

Corresponds to example-glossaries-long.tex

\exampleglossarieslongchild\bib\Corresponds to example-glossaries-longchild.tex

\exampleglossariesmultipar\bib\Corresponds to example-glossaries-multipar.tex

\exampleglossariesparent\bib\Corresponds to example-glossaries-parent.tex

\exampleglossariessymbolnames\bib\Corresponds to example-glossaries-symbolnames.tex

\exampleglossariessymbols\bib\Corresponds to example-glossaries-symbols.tex

\exampleglossariesurl\bib\Corresponds to example-glossaries-url.tex

\exampleglossariesxr\bib\Corresponds to example-glossaries-xr.tex

14.2. Sample Files

The \texttt{glossaries}-extra package comes with some sample files that are listed below. There are also sample files provided with the \texttt{glossaries} package and with \texttt{bib2gls}. See also the Dickimaw Books Gallery.\footnote{dickimaw-books.com/gallery}

\example\sample\tex

Simple sample file that uses one of the dummy files provided by the \texttt{glossaries} package for testing.

\begin{verbatim}
\pdflatex sample
\makeglossaries sample
\pdflatex sample
\pdflatex sample
\end{verbatim}

\example\sampleabbrstyles\tex
14. Supplementary Files

<table>
<thead>
<tr>
<th>Command</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdflatex sample-abbr-styles</td>
<td>makeglossaries sample-abbr-styles</td>
<td>Demonstrates all predefined abbreviation styles.</td>
</tr>
<tr>
<td>pdflatex sample-mixture</td>
<td>makeglossaries sample-mixture</td>
<td>General entries, acronyms and initialisms all treated differently.</td>
</tr>
<tr>
<td>pdflatex sample-name-font</td>
<td>makeglossaries sample-name-font</td>
<td>Categories and attributes are used to customize the way different entries appear.</td>
</tr>
<tr>
<td>pdflatex sample-abbrv</td>
<td>makeglossaries sample-abbrv</td>
<td>General abbreviations.</td>
</tr>
<tr>
<td>pdflatex sample-acronym</td>
<td>makeglossaries sample-acronym</td>
<td>Acronyms aren’t initialisms and don’t expand on first use.</td>
</tr>
</tbody>
</table>
14. Supplementary Files

- sample-acronym-desc.tex

```latex
pdflatex sample-acronym-desc
makeglossaries sample-acronym-desc
pdflatex sample-acronym-desc
makeglossaries sample-acronym-desc
pdflatex sample-acronym-desc
```

Acronyms that have a separate long form and description.

- sample-crossref.tex

```latex
pdflatex sample-crossref
makeglossaries sample-crossref
pdflatex sample-crossref
```

Unused entries that have been cross-referenced automatically are added at the end of the document.

- sample-indexhook.tex

```latex
pdflatex sample-indexhook
makeglossaries sample-indexhook
pdflatex sample-indexhook
```

Use the index hook to track which entries have been indexed (and therefore find out which ones haven’t been indexed).

- sample-footnote.tex

```latex
pdflatex sample-footnote
makeglossaries sample-footnote
pdflatex sample-footnote
```

Footnote abbreviation style that moves the footnote marker outside of the hyperlink generated by `\gls` and moves it after certain punctuation characters for neatness.
14. Supplementary Files

Warn on undefined entries instead of generating an error.

- sample-mixed-abbrv-styles.tex

Different abbreviation styles for different entries.

- sample-initialisms.tex

Automatically insert dots into initialisms.

- sample-postdot.tex

Another initialisms example.

- sample-postlink.tex

Automatically inserting text after the link text produced by commands like \gls (outside of hyperlink, if present).

- sample-header.tex
14. Supplementary Files

Using entries in section/chapter headings.

\texttt{\textasciitilde sample-autoindex.tex}

Using the \texttt{dualindex} and \texttt{indexname} attributes to automatically add glossary entries to the index (in addition to the glossary location list).

\texttt{\textasciitilde sample-autoindex-hyp.tex}

As previous but uses \texttt{hyperref}.

\texttt{\textasciitilde sample-nested.tex}

Using \texttt{\textbackslash gls} within the value of the \texttt{name} key.

\texttt{\textasciitilde sample-entrycount.tex}
14. Supplementary Files

Enable entry-use counting (only index if used more than $n$ times, see §6.1).

[pdflatex sample-entrycount]
[pdflatex sample-entrycount]
[makeglossaries sample-entrycount]
[pdflatex sample-entrycount]

Enable use of per-unit entry-use counting (§6.1).

[pdflatex sample-unitentrycount]
[pdflatex sample-unitentrycount]
[makeglossaries sample-unitentrycount]
[pdflatex sample-unitentrycount]

Using the per-unit entry counting (§6.1) to only have one hyperlink per entry per page.

[pdflatex sample-onelink]
[makeglossaries sample-onelink]
[pdflatex sample-onelink]

Using link counting (§6.2) to only have one hyperlink per entry.

[pdflatex sample-linkcount]
[makeglossaries sample-linkcount]
[pdflatex sample-linkcount]

Insert “page” or “pages” before the location list.

[pdflatex sample-pages]
[makeglossaries sample-pages]
[pdflatex sample-pages]
[pdflatex sample-pages]
[pdflatex sample-pages]

587
14. Supplementary Files

Set the default options for commands like \gls and add an alternative modifier.
- \texttt{sample-mixedsort.tex}

Uses the optional argument of \makeglossaries to allow a mixture of \printglossary and \printnoidxglossary.
- \texttt{sample-external.tex}

Uses the \texttt{targeturl} attribute to allow for entries that should link to an external URL rather than to an internal glossary.
- \texttt{sample-fmt.tex}

Provides text-block commands associated with entries in order to use \glsxtrfmt.
- \texttt{sample-alias.tex}

Uses the \texttt{alias} key (see §3.4).
- \texttt{sample-alttree.tex}
pdflatex sample-alttree
makeglossaries sample-alttree
pdflatex sample-alttree

Uses the glossaries-extra-stylemods package with the alttree style (see §8.6.5).

sample-alttree-sym.tex

pdflatex sample-alttree-sym
makeglossaries sample-alttree-sym
pdflatex sample-alttree-sym

Another alttree example that measures the symbol widths.

sample-alttree-marginpar.tex

pdflatex sample-alttree-marginpar
makeglossaries sample-alttree-marginpar
pdflatex sample-alttree-marginpar

Another alttree example that puts the location list in the margin.

sample-onthefly.tex

pdflatex sample-onthefly
makeglossaries sample-onthefly
pdflatex sample-onthefly

Using on-the-fly commands. Terms with accents must have the name key explicitly set.

sample-onthefly-xetex.tex

xelatex sample-onthefly-xetex
makeglossaries sample-onthefly-xetex
xelatex sample-onthefly-xetex

Using on-the-fly commands with XƎLаЁпX. Terms with UTF-8 characters don’t need to have the name key explicitly set. Terms that contain commands must have the name key explicitly set with the commands removed from the label.

sample-onthefly-utf8.tex
Tries to emulate the previous sample file for use with \LaTeX through the starred version of \GlsXtrEnableOnTheFly. This is a bit iffy and may not always work. Terms that contain commands must have the name key explicitly set with the commands removed from the label.

sample-accsupp.tex

Integrate glossaries-accsupp.

sample-prefix.tex

Integrate glossaries-prefix.

sample-suppl-main.tex

Uses the \texttt{thevalue} to reference a location in the supplementary file sample-suppl.tex.

sample-suppl-main-hyp.tex
14. Supplementary Files

A more complicated version to the above that uses the hyperref package to reference a location in the supplementary file sample-suppl-hyp.tex.
15. Multi-Lingual Support

There’s only one command provided by glossaries-extra that you’re likely to want to change in your document and that’s \abbreviationsname if you use the abbreviations package option to automatically create the glossary labelled abbreviations. If this command doesn’t already exist, it will be defined to “Abbreviations” if babel hasn’t been loaded, otherwise it will be defined as \acronymname (provided by glossaries), which is language-sensitive.

You can redefine \abbreviationsname in the usual way. For example:

\renewcommand*{\abbreviationsname}{List of Abbreviations}

Or using babel or polyglossia captions hook:

\appto\captionsenglish{%
  \renewcommand*{\abbreviationsname}{List of Abbreviations}%
}

Alternatively you can use the title key when you print the list of abbreviations. For example:

\printabbreviations[title={List of Abbreviations}]

or

\printabbreviations[typeabbreviations,title={List of Abbreviations}]

The other fixed text commands are the diagnostic messages, which shouldn’t appear in the final draft of your document.

The glossaries-extra package has the facility to load language modules (whose filename is in the form glossariesxtr-⟨language⟩.ldf) if they exist, but won’t warn if they don’t. If glossaries-extra-bib2gls is loaded via the record package option then the check for language resource files will additionally search for an associated language script file given by glossariesxtr-⟨script⟩.ldf where ⟨script⟩ is the four letter script identifier, such as Latn, associated with the given dialect. There’s no warning if the associated file isn’t found. The
15. Multi-Lingual Support

script file is loaded after the dialect file.

If you want to write your own language module, you just need to create a file called `glossariesxtr-⟨lang⟩.ldf`, where ⟨lang⟩ identifies the language or dialect (see the track-lang package). For example, `glossariesxtr-french.1df`. The file should start with:

\ProvidesGlossariesExtraLang{⟨tag⟩}

The simplest code for this file is:

\ProvidesGlossariesExtraLang{french}[2015/12/09 v1.0]
\newcommand*{\glossariesxtrcaptionsfrench}{%\def\abbreviationsname{Abr\'eviations}\\glossariesxtrcaptionsfrench\\ifcsdef{captions\CurrentTrackedDialect}{%\csappto{captions\CurrentTrackedDialect}{\glossariesxtrcaptionsfrench}\}%\ifcsdef{captions\CurrentTrackedLanguage}{%\csappto{captions\CurrentTrackedLanguage}{\glossariesxtrcaptionsfrench}\}%}\}

You can adapt this for other languages by replacing all instances of the language identifier `french` and the translated text ‘Abr\’eviations’ as appropriate. You can also use the `.ldf` file to provide rule blocks for a particular language for use with `bib2gls`’s custom sort rule. See §11.5 for further details.
15. Multi-Lingual Support

For further information on tracklang, see the tracklang documentation or Localisation with tracklang.tex.\footnote{dickimaw-books.com/latex/tracklang/}

This .ldf file then needs to be put somewhere on \TeX’s path so that it can be found by glossaries-extra. You might also want to consider uploading it to CTAN\footnote{ctan.org/} so that it can be useful to others. (Please don’t send it to me. I already have more packages than I am able to maintain.)

If you additionally want to provide translations for the diagnostic messages used when a glossary is missing, you need to redefine the following commands:

\begin{verbatim}
\GlsXtrNoGlsWarningHead{⟨glossary-label⟩}{⟨file⟩}
\end{verbatim}

This produces the following text in English:

This document is incomplete. The external file associated with the glossary ‘⟨glossary-label⟩’ (which should be called ⟨file⟩) hasn’t been created.

\begin{verbatim}
\GlsXtrNoGlsWarningEmptyStart
\end{verbatim}

This produces the following text in English:

This has probably happened because there are no entries defined in this glossary.

\begin{verbatim}
\GlsXtrNoGlsWarningEmptyMain
\end{verbatim}

This produces the following text in English:

If you don’t want this glossary, add nomain to your package option list when you load glossaries-extra.sty. For example:

\begin{verbatim}
\GlsXtrNoGlsWarningEmptyNotMain{⟨glossary-label⟩}
\end{verbatim}

This produces the following text in English:
Did you forget to use `type=⟨glossary-label⟩` when you defined your entries? If you tried to load entries into this glossary with `\loadglsentries` did you remember to use `[⟨glossary-label⟩]` as the optional argument? If you did, check that the definitions in the file you loaded all had the type set to `\glsdefaulttype`.

```
\GlsXtrNoGlsWarningCheckFile{⟨file⟩}
```

This produces the following text in English:

> Check the contents of the file `⟨file⟩`. If it’s empty, that means you haven’t indexed any of your entries in this glossary (using commands like `\gls` or `\glsadd`) so this list can’t be generated. If the file isn’t empty, the document build process hasn’t been completed.

```
\GlsXtrNoGlsWarningMisMatch
```

This produces the following text in English:

> You need to either replace `\makenoidxglossaries` with `\makeglossaries` or replace `\printglossary` (or `\printglossaries`) with `\printnoidxglossary` (or `\printnoidxglossaries`) and then rebuild this document.

```
\GlsXtrNoGlsWarningNoOut{⟨file⟩}
```

This produces the following text in English:

> The file `⟨file⟩` doesn’t exist. This most likely means you haven’t used `\makeglossaries` or you have used `\nofiles`. If this is just a draft version of the document, you can suppress this message using the `nomissingglstext` package option.

```
\GlsXtrNoGlsWarningTail
```

This produces the following text in English:

> This message will be removed once the problem has been fixed.

```
\GlsXtrNoGlsWarningBuildInfo
```

595
This is advice on how to generate the glossary files.

\GlsXtrNoGlsWarningAutoMake\{glossary-label\}

This is the message produced when the `automake` option is used, but the document needs a rerun or the shell escape setting doesn't permit the execution of the external application. This command also generates a warning in the transcript file.

See the documented code (`glossaries-extra-code.pdf`) for further details.
Part II.

Summaries and Index
## Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>№</td>
<td>A counter is being described.</td>
</tr>
<tr>
<td>⚱️</td>
<td>The syntax and usage of a command, environment or option etc.</td>
</tr>
<tr>
<td>⏹️</td>
<td>A command, environment or option that is now deprecated.</td>
</tr>
<tr>
<td>⚩️</td>
<td>A command, environment or option that should not be used with glossaries-extra.</td>
</tr>
<tr>
<td>📘</td>
<td>An important message.</td>
</tr>
<tr>
<td>📘</td>
<td>Prominent information.</td>
</tr>
<tr>
<td>📘</td>
<td>\LaTeX{} code to insert into your document.</td>
</tr>
<tr>
<td>📘</td>
<td>The definition of an option value.</td>
</tr>
<tr>
<td>📘</td>
<td>Problematic code which should be avoided.</td>
</tr>
<tr>
<td>📘</td>
<td>How the example code should appear in the PDF.</td>
</tr>
<tr>
<td>📘</td>
<td>An option that takes a value.</td>
</tr>
<tr>
<td>📘</td>
<td>A command-line application invocation that needs to be entered into a terminal or command prompt. See also “Incorporating makeglossaries or makeglossaries-lite or bib2gls into the document build”.</td>
</tr>
<tr>
<td>📘</td>
<td>A boolean option that is initially false.</td>
</tr>
<tr>
<td>📘</td>
<td>A boolean option that is initially true.</td>
</tr>
<tr>
<td>📘</td>
<td>Text in a transcript or log file or written to STDOUT or STDERR.</td>
</tr>
<tr>
<td>📘</td>
<td>Code that requires a native Unicode engine (\LaTeX{} or Lua\LaTeX{}).</td>
</tr>
<tr>
<td>📘</td>
<td>An option that doesn’t take a value.</td>
</tr>
<tr>
<td>📘</td>
<td>A warning.</td>
</tr>
</tbody>
</table>

---

*dickimaw-books.com/latex/buildglossaries*
Terms

American Standard Code for Information Interchange (ASCII)
A single-byte character encoding. Related blog article: Binary Files, Text Files and File Encodings.¹

Category post-description hook
A post-description hook called \glsxtrpostdesc\{category\} that is associated with a particular category.

Category post-link hook
A post-link hook called \glsxtrpostlink\{category\} that is associated with a particular category.

Category post-name hook
A post-name hook called \glsxtrpostname\{category\} that is associated with a particular category.

Command-line interface (CLI)
An application that doesn’t have a graphical user interface. That is, an application that doesn’t have any windows, buttons or menus and can be run in a command prompt or terminal.²

Display full form
The full form of an abbreviation obtained with the first use of \gls or \glspl (or case-changing variants). This may or may not produce the same result as the inline full form the corresponding \glsxtrfull command, depending on the abbreviation style.

Entry location
The location of the entry in the document (obtained from the location counter or from the \thevalue option). This defaults to the page number on which the entry has been referenced with any of the \gls-like, \glstext-like or \glsadd commands. An entry may have multiple locations that form a list.

Field
Entry data is stored in fields. These may have a corresponding key used to set the value, such as name or description, but field values may also be set using commands like \GlsXtrSetField, in which case there doesn’t need to be a corresponding key. Some fields are

¹dickimaw-books.com/blog/binary-files-text-files-and-file-encodings/
²dickimaw-books.com/latex/novices/html/terminal.html
considered internal fields. If you are using \bib2gls, it will only recognise fields in the \bib file that have had a key defined in the document or that are special to \bib2gls.

**First use**
The first time an entry is used by a command that unsets the first use flag (or the first time since the flag was reset). For multi-entry sets, see first use.

**First use flag**
A conditional that keeps track of whether or not an entry has been referenced by any of the \gls-like commands (which can adjust their behaviour according to whether or not this flag is true). The conditional is true if the entry hasn’t been used by one of these commands (or if the flag has been reset) and false if it has been used (or if the flag has been unset). Note that multi-entries have their own flag that’s distinct from the first use flags of the individual elements.

**First use text**
The link text that is displayed on first use of the \gls-like commands.

**Group (letters, numbers, symbols)**
A logical division within a glossary that is typically a by-product of the indexing application’s sorting algorithm. See also Gallery: Logical Glossary Divisions (type vs group vs parent).

**Graphical user interface (GUI)**
An application that has windows, buttons or menus.

**Glossary**
Technically a glossary is an alphabetical list of words relating to a particular topic. For the purposes of describing the glossaries and glossaries-extra packages, a glossary is either the list produced by commands like \printglossary or \printunsrtglossary (which may or may not be ordered alphabetically) or a glossary is a set of entry labels where the set is identified by the glossary label or type.

**\gls-like**
Commands like \gls and \glsdisp that change the first use flag. These commands index the entry (if indexing is enabled), create a hyperlink to the entry’s glossary listing (if enabled) and unset the first use flag. These commands end with the post-link hook.

**\glistext-like**
Commands like \glistext and \glslink that don’t change the first use flag. These commands index the entry (if indexing is enabled) and create a hyperlink to the entry’s glossary listing (if enabled). These commands end with the post-link hook.

**Hierarchical level**
A number that indicates how many ancestors an entry has. An entry with no parent has hierarchical level 0. If an entry has a parent then the hierarchical level for the entry is one.

---

1[dickimaw-books.com/gallery/index.php?label=logicaldivisions]
more than the hierarchical level of the parent. Most styles will format an entry according
to its hierarchical level, giving prominence to level 0 entries, although some may have a
maximum supported limit. The “unsrt” family of commands allow the hierarchical level to be
locally adjusted by adding an offset (leveloffset) or to have hierarchy ignored (flatten).

**Ignored glossary**

A glossary that has been defined using a command like \newignoredglossary. These
glossaries are omitted by iterative commands, such as \printglossaries and \printunsrtglossaries. An ignored glossary can only be displayed with \printunsrtglossary
or \printunsrtinnerglossary.

**Ignored location (or record)**

A location that uses glsignore as the encap. With bib2gls, this indicates that the entry
needs to be selected but the location isn’t added to the location list. With other methods, this
will simply create an invisible location, which can result in unwanted commas if the location
list has other items.

**Indexing application**

An application (piece of software) separate from \TeX/\LaTeX that collates and sorts information
that has an associated page reference. Generally the information is an index entry but in this
case the information is a glossary entry. There are two main indexing applications that are
used with \TeX: \makeindex and \xindy. (There is also a new application called \xindex, but
this isn’t supported by glossaries or glossaries-extra.) The glossaries-extra package addition-
ally supports bib2gls. These are all CLI applications.

**Indexing (or recording)**

The process of saving the entry location and any associated information that is required in
the glossary. In the case of makeindex and \xindy, the entry location, encap, entry item
and sort value are written to a supplementary file associated with the glossary that is sub-
sequently read by makeindex/\xindy. In the case of bib2gls and the “noidx” method, the
entry location, encap and label is written to the aux file. With record=nameref, the cur-
rent title and hyperlink target are also included. With bib2gls, each line of data in the aux
is referred to as a “record” and indexing is referred to as “recording”. The record=hybrid
option indexes twice: a bib2gls record in the aux file and a makeindex/\xindy line in the
corresponding file, See §5.8.

**Inline full form**

The full form of an abbreviation obtained with \glsxtrfull or \glsxtrfullpl (or case-
changing variants). This may or may not produce the same result as the full form on first use
of the corresponding \gls-like command (the display full form), depending on the abbrevi-
aption style.

**Inner formatting**

The formatting applied by the inntertextformat option, which redefines \glsxtrgenentry-
textfmt. The inner formatting can only be applied if \glsxtrgenentrytextfmt is embed-
ded within the entry’s display style.
Terms

**Internal field**
An internal field may refer to a key that shouldn't be used in the bib file (internal field (bib2gls)), such as the group field, or it may refer to the label used to internally represent the field (which may or may not match the key used to set the field or may not have an associated key), such as useri which corresponds to the user1 key, or it may refer to a field that is only ever used internally that should not be explicitly modified, such as the field used to store the entry’s hierarchical level.

**Internal field (bib2gls)**
A field that is used or assigned by bib2gls that should typically not be used in the bib file.

**Internal field label**
The field label that forms part of the internal control sequence used to store the field value. This may or may not match the key used to assign the value when defining the entry. See the "Key to Field Mappings" table in the glossaries user manual.

**Link text**
The text produced by \gls-like and \glstext-like commands that have the potential to be a hyperlink.

**Location counter**
The counter used to obtain the entry location.

**Location encap (format)**
A command used to encapsulate an entry location. The control sequence name (without the leading backslash) is identified by the format key. The default encap is \glstextnumberformat.

**Location list**
A list of entry locations (also called a number list). May be suppressed for all glossaries with the package option nonumberlist or for individual glossaries with nonumberlist. With bib2gls, the list may also be suppressed with save-locations=false.

**Multi-entry first use**
The first time a multi-entry is referenced (or the first time since the multi-entry first use flag was reset). This is not necessary the first use of any of the individual entries that form the set.

**Multi-entry first use flag**
A conditional that keeps track of whether or not a multi-entry has been used. This is distinct from the first use flags of the individual elements.

**Multi-entry subsequent use**
When a multi-entry that has already been marked as used is referenced. This is not necessary the subsequent use of any of the individual entries that form the set.
**Post-description hook**

A hook (\glspostdescription) included in some glossary styles that is used after the description is displayed. The glossaries-extra package modifies this command to provide additional hooks, including category post-description hooks. The glossaries-extra-stylemods package modifies the predefined styles provided with glossaries to ensure that they all use \glspostdescription to allow the category post-description hooks to be implemented.

**Post-link hook**

A hook (command) that is used after link text to allow code to be automatically added. The base glossaries package provides a general purpose hook \glspostlinkhook. The glossaries-extra package modifies this command to provide additional hooks, including category post-link hooks.

**Post-name hook**

A hook (command) that is used after the name is displayed in glossary styles. These hooks are implemented by \glossentryname, which needs to be present in the glossary style. The main hook is \glsxtrpostnamehook, which implements auto-indexing (see §12), performs a general purpose hook \glsxtrapostnamehook and a category-specific hook \glsxtrpostname\langle category\rangle.

**Print “unsrt” glossary commands (and environment)**

The set of commands used for displaying a glossary or partial glossary that have “unsrt” in the name: \printunsrtglossary, \printunsrtglossaries (which internally uses \printunsrtglossary), and \printunsrtinnerglossary. These all simply iterate over the list of entries associated with the given glossary, in the order in which they were added to the glossary (hence “unsrt”, which is short for “unsorted”). The way that bib2gls works is that it sorts the entries according to the resource options and adds the entries to the glossary in the required order. These commands may be used with or without bib2gls. If you don’t use bib2gls, you will need to manually ensure that the entries are added in the desired order. The printunsrtglossarywrap environment may also be included in this category, although it only sets up the start and end of the glossary for use with \printunsrtinnerglossary.

**Resource file**

The glstex file created by bib2gls and loaded by \GlsXtrLoadResources (or \glsxtrresourcefile).

**Resource set**

All the settings (resource options) and entries associated with a particular instance of \GlsXtrLoadResources (or \glsxtrresourcefile).

**Standalone entry**

Normally the link text target is created in the glossary, where the entry’s name and description (and optionally other information, such as the symbol) are shown. It may be that you don’t want a list but need to have the name and description somewhere in the document text.
These are standalone entries. Whilst it is possible to simply use `\glsentryname` and `\glsentrydesc`, the `glossaries-extra` package provides a way inserting the name that includes the `hypertarget` for the link text and obeys the post-name hook.

**Subsequent use**
Using an entry that unsets the first use flag when it has already been unset.

**Unicode Transformation Format (8-bit) (UTF-8)**
A variable-width encoding that uses 8-bit code units. This means that some characters are represented by more than one byte. XƎL and LuaЛ treat the multi-byte sequence as a single token, but the older LTEX formats have single-byte tokens, which can cause complications, although these have mostly been addressed with the newer kernels introduced over the past few years. Related blog article: Binary Files, Text Files and File Encodings.

**Whatsit**
A command whose execution is delayed or an OS-specific special command. This includes writing to external files (which is what indexing does).

\footnote{dickimaw-books.com/blog/binary-files-text-files-and-file-encodings/}
Glossary Entry Keys Summary

These are options that can be passed to commands that define entries, such as `\newglossary`-`entry` or `\newabbreviation`.

access={⟨text⟩}  \(\leadsto\) (requires accsupp)

Accessibility text corresponding to the name field. This field will be automatically set by `\newabbreviation`, if not provided and the nameshortaccess attribute is set. See §9.1.

alias={⟨xr-label⟩}  \(\leadsto\) glossaries-extra

Behaves in a similar manner to `see={\[
\text{\seealsoname}\}(xr-label)}` but also sets up aliasing which makes the link text hyperlink to ⟨xr-label⟩ instead.

category={⟨category-label⟩}  \(\leadsto\) initial: general glossaries-extra

The entry’s category (must be a simple label).

counter={⟨counter-name⟩}  \(\leadsto\)

If set, the value indicates the location counter to use by default when indexing this entry (overrides the counter associated with the glossary or the counter package option).

description={⟨text⟩}  \(\leadsto\)

The entry’s description, as displayed in the glossary. If required in the text, use `\glsdesc` (if indexing and hyperlinks are required) or `\glsentrydesc`. Glossary styles should use `\glossentrydesc` and `\glspostdescription` to incorporate the post-description hook.
Accessibility text corresponding to the `description` field.

The plural form of the entry’s description, if applicable. If omitted, this is set to the same value as the `description`, since descriptions tend not to be a singular entity.

Accessibility text corresponding to the `descriptionplural` field.

The entry’s text, as displayed on first use of \gls-like commands. Note that using an abbreviation style or post-link hooks is a more flexible approach. If omitted, this value is assumed to be the same as the `text` key.

Accessibility text corresponding to the `first` field. This field will be automatically set by \newabbreviation, if not provided and the `firstshortaccess` attribute is set. See §9.1.

The entry’s plural form, as displayed on first use of plural \gls-like commands, such as \glspl. If this key is omitted, then the value will either be the same as the `plural` field, if the `first` key wasn’t used, or the value will be taken from the `first` key with \glsplural-suffix appended.
Glossary Entry Keys Summary

Accessibility text corresponding to the firstplural field. This field will be automatically set by \newabbreviation, if not provided and the firstshortaccess attribute is set. See §9.1.

\begin{Verbatim}
\texttt{group=\{(group-label)\}}
\end{Verbatim}

The group label that identifies which group the entry belongs to. This key is only available with the record=only and record=nameref options, and is set by bib2gls, if invoked with --group or -g. This is an internal key assigned by bib2gls as a by-product of sorting. Explicit use without reference to the order of entries can result in fragmented groups. The corresponding title can be set with \glsxtrsetgrouptitle, although this is more commonly done implicitly within the glstex file.

\begin{Verbatim}
\texttt{location=\{(location-list)\}}
\end{Verbatim}

The formatted location list used by the “unsrt” family of commands. This key is only available with the record option and is set by bib2gls unless save-locationsfalse is set.

\begin{Verbatim}
\texttt{long=\{(long-form)\}}
\end{Verbatim}

A field that is set by \newabbreviation to the entry’s long (unabbreviated) form. It typically shouldn’t be used explicitly with \newglossaryentry as \newabbreviation makes other modifications to ensure that when the entry is referenced with the \gls-like commands, it will obey the appropriate abbreviation style. If you are using bib2gls then this field should be used in the bib file when defining abbreviations.

\begin{Verbatim}
\texttt{longaccess=\{(text)\}}
\end{Verbatim}

\texttt{(requires accsupp)}

Accessibility text corresponding to the long field.

\begin{Verbatim}
\texttt{longplural=\{(long-form)\}}
\end{Verbatim}

As long but the plural form.
Glossary Entry Keys Summary

longpluralaccess={⟨text⟩}  \ \ (requires accsupp)

Accessibility text corresponding to the longplural field.

name={⟨text⟩}  \ \ 

The entry’s name, as displayed in the glossary. This typically isn’t used outside of the glossary (the text and plural keys are used instead). However, if there is a need to specifically display the entry name, use \glsname (if indexing and hyperlinks are required) or \glsentryname. Glossary styles should use \glossentryname, which uses \glsentryname and incorporates the post-name hooks and related attributes.

parent=⟨parent-label⟩  \ \ 

The label of the entry’s parent (from which the entry’s hierarchical level is obtained).

plural={⟨text⟩}  \ \ 

The entry’s plural form, as displayed on subsequent use of plural \gls-like commands, such as \glspl. This should be the appropriate plural form of the value provided by the text key. If omitted, this value is assumed to be the value of the text key with \glspluralsuffix appended.

pluralaccess={⟨text⟩}  \ \ (requires accsupp)

Accessibility text corresponding to the plural field. This field will be automatically set by \newabbreviation, if not provided and the textshortaccess attribute is set. See §9.1.

prefix={⟨text⟩}  \ \ glossaries-prefix v3.14a+

The subsequent use singular prefix.
## Glossary Entry Keys Summary

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>prefixfirst={⟨text⟩}</code></td>
<td>The first use singular prefix.</td>
</tr>
<tr>
<td><code>prefixfirstplural={⟨text⟩}</code></td>
<td>The first use plural prefix.</td>
</tr>
<tr>
<td><code>prefixplural={⟨text⟩}</code></td>
<td>The subsequent use plural prefix.</td>
</tr>
<tr>
<td><code>see={[[⟨tag⟩]⟨xr-list⟩]}</code></td>
<td>With the base glossaries package this simply triggers an automatic cross-reference with \gls-see. The glossaries-extra package additionally saves the value. Use <code>autoseeindex=false</code> to prevent the automatic cross-reference. The <code>⟨tag⟩</code> defaults to <code>\seename</code> and <code>⟨xr-list⟩</code> should be a comma-separated list of entries that have already been defined.</td>
</tr>
<tr>
<td><code>seealso={⟨xr-list⟩}</code></td>
<td>Behaves in a similar manner to <code>see={[[\sealsoname]⟨xr-list⟩]}</code>.</td>
</tr>
<tr>
<td><code>short={⟨short-form⟩}</code></td>
<td>A field that is set by <code>\newabbreviation</code> (and <code>\newacronym</code>) to the entry’s short (abbreviated) form. It typically shouldn’t be used explicitly with <code>\newglossaryentry</code> as <code>\newabbreviation</code> makes other modifications to ensure that when the entry is referenced with the \gls-like commands, it will obey the appropriate abbreviation style. If you are using bib2gls then this field should be used in the bib file when defining abbreviations.</td>
</tr>
<tr>
<td><code>shortaccess={⟨text⟩}</code></td>
<td>(requires <code>accsupp</code>)</td>
</tr>
</tbody>
</table>

---

609
Accessibility text corresponding to the `short` field. This field will be automatically set by `\newabbreviation`, if not provided. See §9.1.

```
shortplural={⟨short-form⟩}
```

As `short` but the plural form. The default is obtained by appending the abbreviation plural suffix, but this behaviour can be altered by category attributes. See §4 for further details.

```
shortpluralaccess={⟨text⟩}  (requires accsupp)
```

Accessibility text corresponding to the `shortplural` field. This field will be automatically set by `\newabbreviation`, if not provided. See §9.1.

```
sort={⟨value⟩}  initial: ⟨entry name⟩
```

Specifies the value to use for sorting (overrides the default). This key is usually required for xindy if the `name` key only contains commands (for example, the entry is a symbol), but explicitly using this key in other contexts can break certain sort methods. Don’t use the `sort` field with `bib2gls`.

```
symbol={⟨symbol⟩}  initial: \relax
```

The entry’s associated symbol (optional), which can be displayed with `\glsymbol` (if indexing and hyperlinks are required) or with `\glsentrysymbol`.

```
symbolaccess={⟨text⟩}  (requires accsupp)
```

Accessibility text corresponding to the `symbol` field.

```
symbolplural={⟨symbol plural⟩}
```

\footnote{\url{dickimaw-books.com/gallery/index.php?label=bib2gls-sorting}}
### Glossary Entry Keys Summary

The plural form of the symbol, if applicable, which can be displayed with \texttt{\glsymbolplural} (if indexing and hyperlinks are required) or with \texttt{\glsentrysymbolplural}. If omitted, this value is set to the same as the symbol key (since symbols usually don’t have a plural form).

\[
\begin{array}{|l|}
\hline
\text{symbolpluralaccess} = \{ \langle \text{text} \rangle \} \\
\hline
\end{array}
\]

Accessibility text corresponding to the symbolplural field.

\[
\begin{array}{|l|}
\hline
\text{text} = \{ \langle \text{text} \rangle \} \\
\hline
\end{array}
\]

The entry’s text, as displayed on subsequent use of \texttt{\gls}-like commands. If omitted, this value is assumed to be the same as the name key.

\[
\begin{array}{|l|}
\hline
\text{textaccess} = \{ \langle \text{text} \rangle \} \\
\hline
\end{array}
\]

Accessibility text corresponding to the text field. This field will be automatically set by \texttt{\newabbreviation}, if not provided and the \texttt{textshortaccess} attribute is set. See §9.1.

\[
\begin{array}{|l|}
\hline
\text{type} = \{ \langle \text{glossary-label} \rangle \} \quad \text{initial: \texttt{\glsdefaulttype}} \\
\hline
\end{array}
\]

Assigns the entry to the glossary identified by \langle \text{glossary-label} \rangle.

\[
\begin{array}{|l|}
\hline
\text{user1} = \{ \langle \text{text} \rangle \} \\
\hline
\end{array}
\]

A generic field, which can be displayed with \texttt{\glssuseri} (if indexing and hyperlinks are required) or with \texttt{\glsentryuseri}.

\[
\begin{array}{|l|}
\hline
\text{user1access} = \{ \langle \text{text} \rangle \} \\
\hline
\end{array}
\]

Accessibility text corresponding to the user1 field.
A generic field, which can be displayed with `\glsuserii` (if indexing and hyperlinks are required) or with `\glsentryuserii`.

Accessibility text corresponding to the `user2` field.

A generic field, which can be displayed with `\glsuseriii` (if indexing and hyperlinks are required) or with `\glsentryuseriii`.

Accessibility text corresponding to the `user3` field.

A generic field, which can be displayed with `\glsuseriv` (if indexing and hyperlinks are required) or with `\glsentryuserv`.

Accessibility text corresponding to the `user4` field.

A generic field, which can be displayed with `\glsuserv` (if indexing and hyperlinks are required) or with `\glsentryuserv`.
Glossary Entry Keys Summary

\texttt{user5access=\{(text)\}} \glossaries-accsupp v4.45+

Accessibility text corresponding to the \texttt{user5} field.

\texttt{user6=\{(text)\}}

A generic field, which can be displayed with \texttt{\glsuservi} (if indexing and hyperlinks are required) or with \texttt{\glsentryuservi}.

\texttt{user6access=\{(text)\}} \glossaries-accsupp v4.45+

Accessibility text corresponding to the \texttt{user6} field.
Glossary Entry Fields Summary

These are internal fields that don’t have a corresponding key.

childcount={\langle number\rangle}

Used with the \texttt{save-child-count} resource option to store the entry’s child count.

childlist={\langle entry-label-list\rangle}

Used with the \texttt{save-child-count} resource option to store the entry’s children as an etoolbox internal list.

indexcounter={\langle target-name\rangle}

Used with the \texttt{indexcounter} package option and the \texttt{save-index-counter} resource option. The value is set to the hyperlink target of the first \texttt{wrglossary} location or the first instance for a specific location encap.

loclist={\langle etoolbox list\rangle}

Used by \texttt{\printnoidxglossary} to provide the locations. The value is an etoolbox list of individual locations which are obtained from the aux file. This field will also be used by the “unsrt” family of commands if \texttt{location} isn’t set.

recordcount={\langle number\rangle}

Used with the --record-count switch to store the total number of records for the associated entry.
**Glossary Entry Fields Summary**

\[ \text{recordcount.}\langle \text{counter}\rangle=\{\langle \text{number}\rangle\} \]

Used with the --record-count switch to store the total number of records with the location counter \(\langle \text{counter}\rangle\) for the associated entry.

\[ \text{recordcount.}\langle \text{counter}\rangle.\langle \text{location}\rangle=\{\langle \text{number}\rangle\} \]

Used with the --record-count-unit switch to store the total number of records with the location counter \(\langle \text{counter}\rangle\) set to \(\langle \text{location}\rangle\) for the associated entry.

\[ \text{record.}\langle \text{counter}\rangle=\{\langle \text{location}\rangle\} \]

Used with \texttt{\GlsXtrRecordCounter} to store an etoolbox internal list of locations (without encap) corresponding to the given counter.

\[ \text{secondarygroup}=\{\langle \text{group-label}\rangle\} \]

Used by \texttt{bib2gls} to store the group label obtained from the secondary sort.
\gls-like and \glstext-like Options

Summary

Most (but not all) of these options can be used in the optional argument of all the \gls-like, \glstext-like and \glsadd commands.

\begin{itemize}
  \item \texttt{counter=\{counter-name\}}\hspace{1cm} \gls{glossaries}
  \end{itemize}

The location counter.

\begin{itemize}
  \item \texttt{format=\{cs-name\}}\hspace{1cm} \gls{glossaries}
  \end{itemize}

The control sequence name (without the leading backslash) that should be used to encapsulate the entry location.

\begin{itemize}
  \item \texttt{hyper=\{boolean\}} \hspace{1cm} \textit{default: true; initial: true} \gls{glossaries}
  \end{itemize}

Determines whether or not the link text should have a hyperlink (provided hyperlinks are supported).

\begin{itemize}
  \item \texttt{hyperoutside=\{boolean\}} \hspace{1cm} \textit{default: true; initial: true} \gls{glossaries-extra v1.21+}
  \end{itemize}

Determines whether the hyperlink should be inside or outside of \glstextformat.

\begin{itemize}
  \item \texttt{innertextformat=\{csname\}} \hspace{1cm} \textit{initial: glsxtrdefaultentrytextfmt} \gls{glossaries-extra v1.49+}
  \end{itemize}

The name of the control sequence to use for the inner formatting.
\gls{-Like and \glstext{-Like Options Summary}

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Initial</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>local</td>
<td>\texttt{boolean}</td>
<td>\texttt{true}; \texttt{false}</td>
<td>\gls{v3.04}++</td>
</tr>
<tr>
<td>noindex</td>
<td>\texttt{boolean}</td>
<td>\texttt{false}</td>
<td>\gls{extra}</td>
</tr>
<tr>
<td>postunset</td>
<td>\texttt{value}</td>
<td>\texttt{global}</td>
<td>\gls{extra} v1.49++</td>
</tr>
<tr>
<td>prefix</td>
<td>\texttt{link-prefix}</td>
<td></td>
<td>\gls{extra} v1.31++</td>
</tr>
<tr>
<td>prerest</td>
<td>\texttt{value}</td>
<td>\texttt{none}</td>
<td>\gls{extra} v1.49++</td>
</tr>
<tr>
<td>preunset</td>
<td>\texttt{value}</td>
<td>\texttt{none}</td>
<td>\gls{extra} v1.49++</td>
</tr>
<tr>
<td>textformat</td>
<td>\texttt{csname}</td>
<td></td>
<td>\gls{extra} v1.30++</td>
</tr>
</tbody>
</table>

If \texttt{true} use \texttt{\glslocalunset} to unset the first use flag, otherwise use \texttt{\glsunset} (only applies to \texttt{\gls{-like commands}).}

If \texttt{true} this option will suppress indexing. If you are using \texttt{bib2gls}, you may want to consider using \texttt{format=glsignore} to prevent a location but ensure that the entry is selected.

Determines whether or not to unset the first use flag after the link text. The value may be one of: \texttt{global}, \texttt{local} or \texttt{none} (only applies to \texttt{\gls{-like commands}).}

The prefix to use for the entry’s hyperlink target.

Determines whether or not to reset the entry before the link text. Allowed values: \texttt{none} (no reset), \texttt{local} (localise the reset) and \texttt{global}.

Determines whether or not to unset the entry before the link text. Allowed values: \texttt{none} (no unset), \texttt{local} (localise the unset) and \texttt{global}.

§5.1.2; 193

§5.1.2; 191

§5.1.2; 196

§5.1.2; 191

§5.1.2; 192

§5.1.2; 190

617
The name of the control sequence to use instead of \glstextformat to encapsulate the link text.

\begin{verbatim}
\newcommand{\theHvalue}{\textit{the-H-value}}
\end{verbatim}

Set the hyper location to this value instead of obtaining it from \theH\textit{counter}.

\begin{verbatim}
\newcommand{\theValue}{\textit{location}}
\end{verbatim}

Set the location to this value instead of obtaining it from the location counter.

\begin{verbatim}
\newcommand{\types}{\{\textit{glossary list}\}}
\end{verbatim}

Only available with \glsaddall, the value is the list of glossaries to iterate over.

\begin{verbatim}
\newcommand{\wrgloss}{\textit{position} \textit{initial: before}}
\end{verbatim}

Determines whether to do the indexing before or after the link text. Allowed values: before and after.
Multi-Entry Set Options Summary

all=⟨option list⟩  \[\S7.10; 360\]

Options to pass to the \texttt{\textbackslash gls}-like command for each element.

category=⟨category-label⟩  \[\S7.9.6; 359\]

The category to assign to the multi-entry set.

encapmain=⟨value⟩  \[\textit{initial}: \texttt{glsnumberformat}\]  \[\S7.9.2; 355\]

The value to pass to the \texttt{format} option for the main entry.

encapothers=⟨value⟩  \[\textit{initial}: \texttt{glsnumberformat}\]  \[\S7.9.2; 355\]

The value to pass to the \texttt{format} option for the “other” elements.

firstprefix=⟨value⟩  \[\S7.9.4; 357\]

The prefix to use on first use of the multi-entry.

firstskipmain=⟨boolean⟩  \[\textit{default}: \text{true}; \textit{initial}: \text{false}\]  \[\S7.9.5; 358\]

Determines whether or not to skip the main entry on first use.
Multi-Entry Set Options Summary

\texttt{firstskipothers=\langle boolean\rangle \quad default: \text{true}; \quad initial: \text{false}} \quad \S 7.9.5; 358

Determined whether or not to skip the “other” elements on first use.

\texttt{firstsuffix=\langle value\rangle} \quad \S 7.9.4; 357

The suffix to use on first use of the multi-entry.

\texttt{hyper=\langle boolean\rangle \quad default: \text{true}} \quad \S 7.10; 361

Indicates whether or not to use hyperlinks, if supported, for all elements. This option is for use in the optional argument of \texttt{\textbackslash{m}gls} and can be set implicitly with the default behaviour of the * and + modifiers.

\texttt{hyper=\langle value\rangle \quad initial: \text{individual}} \quad \S 7.9.6; 358

Indicates which elements should have hyperlinks, if supported. This option is a multi-entry setting, see §7.9.

\texttt{indexmain=\langle value\rangle \quad default: \text{true}; \quad initial: \text{true}} \quad \S 7.9.1; 355

Indicates if the main element should be indexed, should only be indexed on first use or should not indexed.

\texttt{indexothers=\langle value\rangle \quad default: \text{true}; \quad initial: \text{true}} \quad \S 7.9.1; 355

Indicates if the “other” elements should be indexed, should only be indexed on first use or should not indexed.

\texttt{main=\langle option list\rangle} \quad \S 7.10; 360

Options to pass to the \texttt{\textbackslash{g}ls}-like command for the main entry.
Multi-Entry Set Options Summary

\texttt{mglsopts=\langle option list \rangle} \quad $\S$7.9.6; 359

The default options to pass to commands like \texttt{\mgls}.

\texttt{mpostlink=\langle value \rangle} \quad \textit{default: true; initial: true} \quad $\S$7.9.3; 356

Indicates whether or not the multi-entry post-link hook should be enabled and, if so, whether it should only be enabled on first or subsequent use.

\texttt{mpostlinkelement=\langle value \rangle} \quad \textit{initial: last} \quad $\S$7.9.3; 356

Indicates which post-link hook to use if the multi-entry post-link hook has been enabled.

\texttt{multiunset=\langle value \rangle} \quad \textit{initial: global} \quad $\S$7.10; 361

Indicates whether or not the multi-entry first use flag should be unset.

\texttt{others=\langle option list \rangle} \quad $\S$7.10; 360

Options to pass to the \texttt{\gls}-like command for the “other” elements.

\texttt{postlinks=\langle value \rangle} \quad \textit{initial: none} \quad $\S$7.9.3; 356

Indicates which post-link hooks should be enabled.

\texttt{presetlocal=\langle boolean \rangle} \quad \textit{default: true; initial: false} \quad $\S$7.10; 361

Indicates whether or not the prereset options should have a local or global effect.

\texttt{resetall=\langle boolean \rangle} \quad \textit{default: true; initial: false} \quad $\S$7.10; 361
Multi-Entry Set Options Summary

Indicates whether or not to reset all elements’ first use flag before using \gls.

\begin{verbatim}
resetmain=\langle boolean\rangle  
default: true; initial: false  
\end{verbatim}

Indicates whether or not to reset the main entry’s first use flag before using \gls.

\begin{verbatim}
resetothers=\langle boolean\rangle  
default: true; initial: false  
\end{verbatim}

Indicates whether or not to reset all “other” elements’ first use flag before using \gls.

\begin{verbatim}
setup=\langle option list\rangle  
\end{verbatim}

Multi-entry options that will override any conflicting options already assigned to the multi-entry.

\begin{verbatim}
textformat=\langle value\rangle  
initial: @firstofone  
\end{verbatim}

The control sequence name of the command that should encapsulate the entire content.

\begin{verbatim}
unsetall=\langle boolean\rangle  
default: true; initial: false  
\end{verbatim}

Indicates whether or not to unset all elements’ first use flag before using \gls.

\begin{verbatim}
unsetmain=\langle boolean\rangle  
default: true; initial: false  
\end{verbatim}

Indicates whether or not to unset the main entry’s first use flag before using \gls.

\begin{verbatim}
unsetothers=\langle boolean\rangle  
default: true; initial: false  
\end{verbatim}

Indicates whether or not to unset all “other” elements’ first use flag before using \gls.

622
Multi-Entry Set Options Summary

```
usedprefix=⟨value⟩
```

The prefix to use on subsequent use of the multi-entry.

```
usedskipmain=⟨boolean⟩
       default: true; initial: false
```

Determines whether or not to skip the main entry on subsequent use.

```
usedskipothers=⟨boolean⟩
       default: true; initial: false
```

Determines whether or not to skip the “other” elements on subsequent use.

```
usedsuffix=⟨value⟩
```

The suffix to use on subsequent use of the multi-entry.
Print [Unsrt|noidx] Glossary Options

Summary

Most (but not all) of these options can be used in the optional argument of all the print glossary commands: \printglossary, \printnoidxglossary, \printunsrtglossary and \printunsrtinnerglossary. Some may be used in the optional argument of the printunsrt-glossarywrap environment.

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Initial</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entrycounter</td>
<td>\langle boolean\rangle</td>
<td>true</td>
<td>If true, enable the entry counter.</td>
</tr>
<tr>
<td>flatten</td>
<td>\langle boolean\rangle</td>
<td>true</td>
<td>If true, treats all entries as though they have the same hierarchical level (the value of level-offset). This option is only available for the “unsrt” family of commands and the printunsrt-glossarywrap environment.</td>
</tr>
<tr>
<td>groups</td>
<td>\langle boolean\rangle</td>
<td>true</td>
<td>Enables group formation. This option is only available for the “unsrt” family of commands and the printunsrtglossarywrap environment. Note that no groups will be formed when invoking bib2gls with the default --no-group, regardless of this setting.</td>
</tr>
<tr>
<td>label</td>
<td>\langle label\rangle</td>
<td></td>
<td>Adds \label{⟨label⟩} to the start of the glossary (after the title).</td>
</tr>
<tr>
<td>leveloffset</td>
<td>\langle offset\rangle</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

§8.3; 379

§8.3; 378

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379

§8.3; 379
Set or increment the hierarchical level offset. If \langle offset\rangle starts with ++ then the current offset is incremented by the given amount otherwise the current offset is set to \langle offset\rangle. For example, an entry with a normal hierarchical level of 1 will be treated as though it has hierarchical level \(1 + \langle offset\rangle\). This option is only available for the “unsrt” family of commands and the printunsrtglossarywrap environment.

\begin{verbatim}
  nogroupskip=\langle boolean \rangle 
  default: true; initial: false
\end{verbatim}

Suppress the gap implemented by some glossary styles between groups.

\begin{verbatim}
  nonumberlist=\langle boolean \rangle 
  default: true; initial: false
\end{verbatim}

Suppress the location list. Note that nonumberlist=false will have no effect with the save -locationsfalse resource option as there won’t be any location lists to display.

\begin{verbatim}
  nopostdot=\langle boolean \rangle 
  default: true; initial: false
\end{verbatim}

Suppress the post-description punctuation.

\begin{verbatim}
  numberedsection=\langle value \rangle 
  default: nolabel; initial: false
\end{verbatim}

Indicates whether or not glossary section headers will be numbered and also if they should automatically be labelled. The numberedsection package option will change the default setting to match.

\begin{verbatim}
  postamble=\langle text \rangle 
  \glossaries-extra v1.49+
\end{verbatim}

Redefines \glossarypostamble to \langle text\rangle.

\begin{verbatim}
  preamble=\langle text \rangle 
  \glossaries-extra v1.49+
\end{verbatim}

Redefines \glossarypreamble to \langle text\rangle.
prefix=⟨prefix⟩  glossaries-extra v1.31+  §8.3; 379

Redefines \glolinkprefix to ⟨prefix⟩.

sort=⟨method⟩  §8.3; 377

Only available with \printnoidxglossary, this indicates how the glossary should be ordered.

style=⟨style-name⟩  §8.3; 378

Use the ⟨style-name⟩ glossary style.

subentrycounter=⟨boolean⟩  default: true; initial: false  glossaries v4.08+  §8.3; 381

If true, enable the sub-entry counter.

target=⟨boolean⟩  default: true; initial: true  glossaries-extra v1.12+  §8.3; 382

If true, each entry in the glossary should have a hypertarget created, if supported by the glossary style and if hyperlinks are enabled.

targetnameprefix=⟨prefix⟩  glossaries-extra v1.20+  §8.3; 381

Inserts ⟨prefix⟩ at the start of the hypertarget names.

title=⟨text⟩  §8.3; 378

Sets the glossary title (overriding the default).
Sets the glossary toc title (overriding the default).

Identifies the glossary to display.
Abbreviation Styles Summary

Abbreviations defined using \newabbreviation will follow the style associated with the entry’s category. If there is no style associated with the entry’s category, the style for the abbreviation category is used (the default is long–short). Note that glossaries-extra re-defines \newacronym to use \newabbreviation with category=acronym so any entry defined with \newacronym will use the abbreviation style for the acronym category (the default is short–nolong).

- footnote-desc alias: short-footnote-desc
- footnote-em alias: short-em-footnote
- footnote-sc alias: short-sc-footnote
- footnote-sm alias: short-sm-footnote
- footnote alias: short-footnote
- long-desc-em alias: long-noshort-em-desc
- long-desc-sc alias: long-noshort-sc-desc
### Abbreviation Styles Summary

- **long-desc-sm**
  
  alias: long-noshort-sm-desc

- **long-desc**
  
  alias: long-noshort-desc

- **long-em-noshort-em-desc-noreg**

  An abbreviation style like `long-em-noshort-em-desc` but sets the regular attribute to false.

- **long-em-noshort-em-desc**

  An abbreviation style like `long-noshort-desc` but formats both the long and short form in an emphasized font (`\emph`).

- **long-em-noshort-em-noreg**

  An abbreviation style like `long-em-noshort-em` but sets the regular attribute to false.

- **long-em-noshort-em**

  An abbreviation style like `long-noshort` but formats both the long and short form in an emphasized font (`\emph`).

- **long-em-short-em-desc**

  An abbreviation style like `long-short-desc` but formats both the long and short form in an emphasized font (`\emph`).
Abbreviation Styles Summary

**long-em-short-em**

An abbreviation style like long-short but formats both the long and short form in an emphasized font (\textit{}).

**long-em**  
alias: long-noshort-em

**long-hyphen-noshort-desc-noreg**

An abbreviation style like long-noshort-desc but checks if the inserted material starts with a hyphen (use with markwords attribute).

**long-hyphen-noshort-noreg**

An abbreviation style like long-hyphen-short-hyphen but doesn’t show the short form on first use.

**long-hyphen-postshort-hyphen-desc**

An abbreviation style like long-hyphen-short-hyphen-desc but places the insert and parenthetical material in the post-link hook.

**long-hyphen-postshort-hyphen**

An abbreviation style like long-hyphen-short-hyphen but places the insert and parenthetical material in the post-link hook.

**long-hyphen-short-hyphen-desc**

An abbreviation style like long-hyphen-short-hyphen but the description must be supplied.
Abbreviation Styles Summary

- **long-hyphen-short-hyphen**

An abbreviation style like `long-short` but checks if the inserted material starts with a hyphen (use with `markwords` or `markshortwords` attributes).

- **long-noshort-desc-noreg**

As `long-noshort-desc` but it will set the `regular` attribute to false.

- **long-noshort-desc**

An abbreviation style that only shows the long form on first use and subsequent use. The short form won’t be showed unless you use a command like \glsxtrshort. The `description` key must be supplied. The full form will only be shown with commands like \glsxtrfull. This style sets the `regular` attribute to true.

- **long-noshort-em-desc**

An abbreviation style like `long-noshort-desc` but formats the short form in an emphasized font (`\emph`).

- **long-noshort-em**

An abbreviation style like `long-noshort` but formats the short form in an emphasized font (`\emph`).

- **long-noshort-noreg**

As `long-noshort` but it will set the `regular` attribute to false.

- **long-noshort-sc-desc**
Abbreviation Styles Summary

An abbreviation style like `long-noshort-desc` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

`long-noshort-sc`

An abbreviation style like `long-noshort` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

`long-noshort-sm-desc`

An abbreviation style like `long-noshort-desc` but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.

`long-noshort-sm`

An abbreviation style like `long-noshort` but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.

`long-noshort`

An abbreviation style that only shows the long form on first use and subsequent use. The short form won’t be showed unless you use a command like \glsxtrshort. The full form will only be shown with commands like \glsxtrfull. This style sets the regular attribute to true.

`long-only-short-only-desc`

An abbreviation style like `long-only-short-only` but the description must be supplied.

`long-only-short-only`

An abbreviation style that only shows the long form on first use and only shows the short form on subsequent use.
Abbreviation Styles Summary

long-only-short-sc-only-desc

An abbreviation style like `long-only-short-sc-only` but the description must be supplied.

long-only-short-sc-only

An abbreviation style like `long-only-short-only` but uses small caps for the short form.

long-postshort-sc-user-desc

An abbreviation style like `long-postshort-sc-user` but the description must be supplied.

long-postshort-sc-user

An abbreviation style like `long-postshort-user` but formats the short form in a small caps font (`\textsc`). The short form should therefore be in lowercase.

long-postshort-user-desc

An abbreviation style like `long-postshort-user` but the description must be supplied.

long-postshort-user

An abbreviation style like `long-short-user` but the parenthetical content is placed in the postlink hook.

谤 long-sc alias: long-noshort-sc

long-short-desc
Abbreviation Styles Summary

As `long-short` but the **description** must be supplied in ⟨*options*⟩.

<table>
<thead>
<tr>
<th>long-short-em-desc</th>
</tr>
</thead>
</table>

An abbreviation style like `long-short-desc` but formats the short form in an emphasized font (\emph).

<table>
<thead>
<tr>
<th>long-short-em</th>
</tr>
</thead>
</table>

An abbreviation style like `long-short` but formats the short form in an emphasized font (\emph).

<table>
<thead>
<tr>
<th>long-short-sc-desc</th>
</tr>
</thead>
</table>

An abbreviation style like `long-short-desc` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

<table>
<thead>
<tr>
<th>long-short-sc</th>
</tr>
</thead>
</table>

An abbreviation style like `long-short` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

<table>
<thead>
<tr>
<th>long-short-sm-desc</th>
</tr>
</thead>
</table>

An abbreviation style like `long-short-desc` but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.

<table>
<thead>
<tr>
<th>long-short-sm</th>
</tr>
</thead>
</table>

An abbreviation style like `long-short` but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.

634
Abbreviation Styles Summary

**long-short-user-desc**
An abbreviation style like `long-short-user` but the description must be supplied.

**long-short-user**
An abbreviation style like `long-short` but includes the value of the field identified by `\gls-xtruserfield` (if set) in the parenthetical content.

**long-short**
An abbreviation style that shows the long form followed by the short form on first use. If the `⟨insert⟩` argument is used with the `\gls-like` or `\glstext-like` commands, it will be placed after the long form on first use. On subsequent use, only the short form is shown (followed by `⟨insert⟩`, if provided). This style sets the `regular` attribute to `false` (which means that the `\gls-like` commands won’t use the `first/firstplural` or `text/plural` values).

**long-sm** *alias: long-noshort-sm*

**long** *alias: long-noshort*

**nolong-short-em**
An abbreviation style like `nolong-short` but formats the short form in an emphasized font (`\emph`).

**nolong-short-noreg**
As `nolong-short` but it will set the `regular` attribute to `false`.
An abbreviation style like `nolong-short` but formats the short form in a small caps font (`\textsc`). The short form should therefore be in lowercase.

An abbreviation style like `nolong-short` but formats the short form in a smaller font (`\text`). The `relsize` package is must be loaded.

As `short-nolong` but the inline full form shows the long form followed by the short form in parentheses.
### Abbreviation Styles Summary

- **short-desc**  
  *alias:* short-nolong-desc  
  An abbreviation style like short-footnote-desc but formats the short form in an emphasized font (\textit{}).

- **short-em-desc**  
  *alias:* short-em-nolong-desc  
  An abbreviation style like short-footnote but formats the short form in an emphasized font (\textit{}).

- **short-em-footnote-desc**  
  An abbreviation style like short-long-desc but formats the short form in an emphasized font (\textit{}).

- **short-em-long-desc**  
  An abbreviation style like short-long-desc but formats both the long and short form in an emphasized font (\textit{}).

- **short-em-long-em-desc**  
  An abbreviation style like short-long-desc but formats both the long and short form in an emphasized font (\textit{}).

- **short-em-long-em**  
  An abbreviation style like short-long but formats both the long and short form in an emphasized font (\textit{}).

- **short-em-long**  
  An abbreviation style like short-long but formats both the long and short form in an emphasized font (\textit{}).
An abbreviation style like short-long but formats the short form in an emphasized font (\textit{emph}).

\begin{itemize}
\item \texttt{short-em-nolong-desc}
\end{itemize}

72, 73

An abbreviation style like short-nolong-desc but formats the short form in an emphasized font (\textit{emph}).

\begin{itemize}
\item \texttt{short-em-nolong}
\end{itemize}

71, 72

An abbreviation style like short-nolong but formats the short form in an emphasized font (\textit{emph}).

\begin{itemize}
\item \texttt{short-em-postfootnote-desc}
\end{itemize}

129

An abbreviation style like short-postfootnote-desc but formats the short form in an emphasized font (\textit{emph}).

\begin{itemize}
\item \texttt{short-em-postfootnote}
\end{itemize}

128

An abbreviation style like short-postfootnote but formats the short form in an emphasized font (\textit{emph}).

\begin{itemize}
\item \texttt{short-em} \textit{alias: short-em-nolong}
\end{itemize}

72

An abbreviation style like short-footnote-desc but formats the short form in an emphasized font (\textit{emph}).

\begin{itemize}
\item \texttt{short-footnote-desc}
\end{itemize}

115, 116

As short-footnote but the description must be supplied in (\textit{options}).

\begin{itemize}
\item \texttt{short-footnote}
\end{itemize}

114, 115

638
Abbreviation Styles Summary

An abbreviation style that shows the short form with the long form as a footnote on first use. If the \( \langle \text{insert} \rangle \) argument is used with the \texttt{\glslike} or \texttt{\glstextlike} commands, it will be placed after the short form, before the footnote marker, on first use. On subsequent use, only the short form is shown (followed by \( \langle \text{insert} \rangle \), if provided). The inline full form shows the short form followed by the long form in parentheses. This style sets the \texttt{regular} attribute to false (which means that the \texttt{\glslike} commands won’t use the \texttt{first/first-plural} or \texttt{text/plural} values). This style also sets the \texttt{nohyperfirst} attribute to true to avoid nesting the footnote marker link. If you want hyperlinks on first use, use the \texttt{short-postfootnote} style instead.

\begin{description}
\item[short-hyphen-long-hyphen-desc] An abbreviation style like \texttt{short-hyphen-long-hyphen} but the description must be supplied.
\item[short-hyphen-long-hyphen] An abbreviation style like \texttt{short-long} but checks if the inserted material starts with a hyphen (use with \texttt{markwords} or \texttt{markshortwords} attributes).
\item[short-hyphen-postlong-hyphen-desc] An abbreviation style like \texttt{short-hyphen-long-hyphen-desc} but the insert and parenthetical material are placed in the post-link hook.
\item[short-hyphen-postlong-hyphen] An abbreviation style like \texttt{short-hyphen-long-hyphen} but the insert and parenthetical material are placed in the post-link hook.
\item[short-long-desc] As \texttt{short-long} but the \texttt{description} must be supplied in \texttt{\langle options \rangle}.
\end{description}
# Abbreviation Styles Summary

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>short-long-user-desc</strong></td>
<td>An abbreviation style like <code>short-long-user</code> but the description must be supplied.</td>
</tr>
<tr>
<td><strong>short-long-user</strong></td>
<td>An abbreviation style like <code>short-long</code> but includes the value of the field identified by \glsxtruserfield (if set) in the parenthetical content.</td>
</tr>
<tr>
<td><strong>short-long</strong></td>
<td>An abbreviation style that shows the short form followed by the long form on first use. If the \textit{insert} argument is used with the \gls-like or \glstext-like commands, it will be placed after the short form on first use. On subsequent use, only the short form is shown (followed by \textit{insert}, if provided). This style sets the regular attribute to false (which means that the \gls-like commands won’t use the first/firstplural or text/plural values).</td>
</tr>
<tr>
<td><strong>short-nolong-desc-noreg</strong></td>
<td>As <code>short-nolong-desc</code> but it will set the regular attribute to false.</td>
</tr>
<tr>
<td><strong>short-nolong-desc</strong></td>
<td>As <code>short-nolong</code> but the description must be supplied in \textit{options}.</td>
</tr>
<tr>
<td><strong>short-nolong-noreg</strong></td>
<td>As <code>short-nolong</code> but it will set the regular attribute to false.</td>
</tr>
<tr>
<td><strong>short-nolong</strong></td>
<td>An abbreviation style that only shows the short form on first use and subsequent use. The long form won’t be showed unless you use a command like \glsxtrlong. The full form will</td>
</tr>
</tbody>
</table>
Abbreviation Styles Summary

only be shown with commands like \glsxtrfull. This style sets the regular attribute to true.

**short-postfootnote-desc**

As **short-postfootnote** but the description must be supplied in \textit{options}.

**short-postfootnote**

Similar to **short-footnote** but the footnote is placed in the post-link hook.

**short-postlong-user-desc**

An abbreviation style like **short-postlong-user** but the description must be supplied.

**short-postlong-user**

An abbreviation style like **short-long** but includes the value of the field identified by \glsxtruserfield (if set) in the parenthetical content, which is placed in the post-link hook.

**short-sc-desc**  
alias: **short-sc-nolong-desc**

**short-sc-footnote-desc**

An abbreviation style like **short-footnote-desc** but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

**short-sc-footnote**

An abbreviation style like **short-footnote** but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.
Abbreviation Styles Summary

**short-sc-long-desc**

An abbreviation style like `short-long-desc` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

**short-sc-long**

An abbreviation style like `short-long` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

**short-sc-nolong-desc**

An abbreviation style like `short-nolong-desc` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

**short-sc-nolong**

An abbreviation style like `short-nolong` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

**short-sc-postfootnote-desc**

An abbreviation style like `short-postfootnote-desc` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

**short-sc-postfootnote**

An abbreviation style like `short-postfootnote` but formats the short form in a small caps font (\textsc). The short form should therefore be in lowercase.

**short-sc**

*alias: short-sc-nolong*
### Abbreviation Styles Summary

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>short-sm-desc</td>
<td>An abbreviation style like short-footnote-desc but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.</td>
<td>71</td>
</tr>
<tr>
<td>short-sm-footnote-desc</td>
<td>An abbreviation style like short-footnote but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.</td>
<td>123</td>
</tr>
<tr>
<td>short-sm-footnote</td>
<td>An abbreviation style like short-footnote but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.</td>
<td>122</td>
</tr>
<tr>
<td>short-sm-long-desc</td>
<td>An abbreviation style like short-long-desc but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.</td>
<td>96</td>
</tr>
<tr>
<td>short-sm-long</td>
<td>An abbreviation style like short-long but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.</td>
<td>96</td>
</tr>
<tr>
<td>short-sm-nolong-desc</td>
<td>An abbreviation style like short-nolong-desc but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.</td>
<td>70, 71</td>
</tr>
<tr>
<td>short-sm-nolong</td>
<td>An abbreviation style like short-nolong but formats the short form in a smaller font (\textsmaller). The relsize package is must be loaded.</td>
<td>69, 70</td>
</tr>
</tbody>
</table>
An abbreviation style like `short-postfootnote-desc` but formats the short form in a smaller font (`\textsmaller`). The `relsize` package is must be loaded.

An abbreviation style like `short-postfootnote` but formats the short form in a smaller font (`\textsmaller`). The `relsize` package is must be loaded.

- **short-sm** \textit{alias: short-sm-nolong}

- **short** \textit{alias: short-nolong}
Glossary Styles Summary

The default style may be set with \setglossarystyle or use:

\usepackage[stylemods=⟨name⟩,style=⟨style-name⟩]{glossaries-extra}

where the style is provided by package glossary-⟨name⟩. The default style can be overridden for individual glossaries with the style option. For a summary of all available styles, see Gallery: Predefined Styles.¹

<table>
<thead>
<tr>
<th>Style</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbr-long-short</td>
<td>glossary-longextra v1.49+</td>
</tr>
<tr>
<td>abbr-short-long</td>
<td>glossary-longextra v1.49+</td>
</tr>
<tr>
<td>altlist</td>
<td>glossary-list</td>
</tr>
<tr>
<td>alttree</td>
<td>glossary-tree</td>
</tr>
<tr>
<td>alttreegroup</td>
<td>glossary-tree</td>
</tr>
</tbody>
</table>

This style displays the glossary in a table for (either longtable or tabular) with two columns: the long form and the short form.

This style displays the glossary in a table for (either longtable or tabular) with two columns: the short form and the long form.

As list but starts the description on a new line.

Like tree but the width of the widest name must be supplied (using a command like \glssetwidest).

¹dickimaw-books.com/gallery/index.php?label=glossaries-styles
As alttree but has headers at the start of each group.

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bookindex</code></td>
<td>This style is designed for indexes. Symbols and descriptions are not shown. Since descriptions aren’t shown, there’s no post-description hook.</td>
</tr>
<tr>
<td><code>index</code></td>
<td>A hierarchical style that supports up to level 2, similar to normal indexes, but symbols and descriptions are shown.</td>
</tr>
<tr>
<td><code>indexgroup</code></td>
<td>As index but has headers at the start of each group.</td>
</tr>
<tr>
<td><code>inline</code></td>
<td>A compact style with all entries listed in the same paragraph and no groups, locations or symbols.</td>
</tr>
<tr>
<td><code>list</code></td>
<td>This style uses the description environment and places the entry name in the optional argument of \item. Symbols and sub-entry names are not shown.</td>
</tr>
<tr>
<td><code>listdotted</code></td>
<td>A list-like style that has a dotted leader between the name and description. The location list isn’t shown.</td>
</tr>
<tr>
<td><code>listgroup</code></td>
<td></td>
</tr>
</tbody>
</table>
Glossary Styles Summary

As list but has headers at the start of each group.

```
listhypergroup
```

As listgroup but has a row at the start with hyperlinks to each group.

```
long-booktabs
```

This style displays the glossary using `longtable` and horizontal rules from the `booktabs` package.

```
long-desc-name
```

This style displays the glossary in a table for (either `longtable` or `tabular`) with two columns: the description and the name.

```
long-desc-sym-name
```

This style displays the glossary in a table for (either `longtable` or `tabular`) with three columns: the description, symbol and name.

```
long-desc-sym
```

This style displays the glossary in a table for (either `longtable` or `tabular`) with two columns: the description and the symbol.

```
long-loc-desc-name
```

This style displays the glossary in a table for (either `longtable` or `tabular`) with three columns: the location list, description and name.

```
long-loc-desc-sym-name
```
This style displays the glossary in a table for (either longtable or tabular) with four columns: the location list, description, symbol and name.
This style displays the glossary in a table for (either longtable or tabular) with three columns: the name, symbol and description.

This style displays the glossary in a table for (either longtable or tabular) with three columns: the symbol, description and name.

This style displays the glossary in a table for (either longtable or tabular) with two columns: the symbol and the description.

This style uses the longtable environment (provided by the longtable package). Symbols and sub-entry names are not shown.

This style displays the glossary using longtable with ragged right paragraph formatting for the description column.

As index but puts the content inside a multicols environment.

As mcolindex but has headers at the start of each group.
Glossary Styles Summary

**mcoltree**  
As tree but puts the content inside a multicols environment.

**super**  
This style displays the glossary using supertabular.

**superragged**  
This style displays the glossary using supertabular with ragged right paragraph formatting for the description column.

**table**  
This style is specific to `\printunsrttable`.

**topic**  
This style is designed for hierarchical glossaries where the top-level entry represents a topic.

**topicmcols**  
Similar to topic but the sub-entries are placed in a multicols environment.

**tree**  
A hierarchical style that supports unlimited levels (although a deep hierarchy may not fit the available line width) with that shows symbols and descriptions.

650
As tree but has headers at the start of each group.

| treehypergroup | glossary-tree |

As treegroup but has a row at the start with hyperlinks to each group.

| treenoname | glossary-tree |

Like tree but the child entries don’t have their name shown.
## Command Summary

### \glsxtr@doglossary

Internal command used within the construction of the glossary code by the “unsrt” family of commands. Should not be used or modified but `\printunsrtglossarypredoglossary` can be defined to show the definition of this command for debugging purposes.

### \glsxtr@multientry{(options)}{(multi-label)}{(main-label)}{(list)}

Information in the `aux` about a multi-label defined in the previous \LaTeX run.

### \glsxtr@org@starttoc{(toc)}

Set to the definition of `\starttoc` when `glossaries-extra` loads.

### \glsxtr@org@markboth{(left text)}{(right text)}

Set to the definition of `\markboth` when `glossaries-extra` loads.

### \glsxtr@org@markright{(text)}

Set to the definition of `\markright` when `glossaries-extra` loads.
Command Summary

Redefines \glsxtrifinmark to do its first argument (\textit{true}).

\texttt{@glsxtrnotinmark}  \hfill glossaries-extra v1.07+

\texttt{\@glsxtrnotinmark}  \hfill §5.3.3; 227

Redefines \glsxtrifinmark to do its first argument (\textit{true}).

\A
\texttt{\AB{\langle options\rangle}{\langle entry-label\rangle}{\langle inset\rangle}}  \hfill §4.3.2; Table 4.1

A synonym for \cGLS defined by the shortcuts=abbreviations package option.

\Ab
\texttt{\Ab{\langle options\rangle}{\langle entry-label\rangle}{\langle inset\rangle}}  \hfill §4.3.2; Table 4.1

A synonym for \cGls defined by the shortcuts=abbreviations package option.

\ab
\texttt{\ab{\langle options\rangle}{\langle entry-label\rangle}{\langle inset\rangle}}  \hfill §4.3.2; Table 4.1

A synonym for \cgls defined by the shortcuts=abbreviations package option.

\texttt{\abbreviationsname}  \hfill §2.1; 9

initial: Abbreviations  glossaries-extra

(language-sensitive)

Expands to the title of the abbreviations glossary. The default is “Abbreviations” or \texttt{\acronym-name} if babel has been detected.

\texttt{\abbrvpluralsuffix}  \hfill 171

initial: \glsxtrabbrvpluralsuffix

Style-sensitive abbreviation suffix. This is the command that’s actually used in the value of the \texttt{shortplural} key when an entry is defined with \texttt{\newabbreviation} (unless suppressed with the \texttt{noshortplural} attribute). This command is redefined by the abbreviation styles
to \glsxtrabbrvpluralsuffix or the style’s custom suffix command (such as \glsxtrscsuffix).

\begin{longtable}{p{0.4\textwidth}p{0.6\textwidth}}
\texttt{\textbackslash ABP\{⟨options⟩\}⟨⟨entry-label⟩⟩⟨⟨inset⟩⟩} & modifiers: * + ⟨alt-mod⟩
\end{longtable}

A synonym for \cGLSp1 defined by the shortcuts=Abbreviations package option.

\begin{longtable}{p{0.4\textwidth}p{0.6\textwidth}}
\texttt{\textbackslash A\textbackslash bp\{⟨options⟩\}⟨⟨entry-label⟩⟩⟨⟨inset⟩⟩} & modifiers: * + ⟨alt-mod⟩
\end{longtable}

A synonym for \cGlspl defined by the shortcuts=Abbreviations package option.

\begin{longtable}{p{0.4\textwidth}p{0.6\textwidth}}
\texttt{\textbackslash a\textbackslash bp\{⟨options⟩\}⟨⟨entry-label⟩⟩⟨⟨inset⟩⟩} & modifiers: * + ⟨alt-mod⟩
\end{longtable}

A synonym for \cGlspl defined by the shortcuts=Abbreviations package option.

\begin{longtable}{p{0.4\textwidth}p{0.6\textwidth}}
\texttt{\textbackslash AC\{⟨options⟩\}⟨⟨entry-label⟩⟩⟨⟨inset⟩⟩} & modifiers: * + ⟨alt-mod⟩
\end{longtable}

A synonym for \cGLS defined by the shortcuts=ac package option.

\begin{longtable}{p{0.4\textwidth}p{0.6\textwidth}}
\texttt{\textbackslash A\textbackslash c\{⟨options⟩\}⟨⟨entry-label⟩⟩⟨⟨inset⟩⟩} & modifiers: * + ⟨alt-mod⟩
\end{longtable}

A synonym for \cGLS or \Gls defined by the shortcuts=ac or shortcuts=Acronyms package option, respectively.

\begin{longtable}{p{0.4\textwidth}p{0.6\textwidth}}
\texttt{\textbackslash ac\{⟨options⟩\}⟨⟨entry-label⟩⟩⟨⟨inset⟩⟩} & modifiers: * + ⟨alt-mod⟩
\end{longtable}

A synonym for \cGLS or \Gls defined by the shortcuts=ac or shortcuts=Acronyms package option, respectively.

\begin{longtable}{p{0.4\textwidth}p{0.6\textwidth}}
\texttt{\textbackslash A\textbackslash C\textbackslash f\{⟨options⟩\}⟨⟨entry-label⟩⟩⟨⟨inset⟩⟩} & modifiers: * + ⟨alt-mod⟩
\end{longtable}

§4.3.2; Table 4.1
Command Summary

A synonym for \GLSxtrfull defined by the \texttt{shortcuts=ac} package option.

\texttt{\Acf\{\langle\text{options}\rangle\}\{\langle\text{entry-label}\rangle\}\{\langle\text{inset}\rangle\} \quad \text{modifiers: } * + \langle\text{alt-mod}\rangle}

§4.3.2; Table 4.1

A synonym for \Glsxtrfull or \texttt{\Acrfull} defined by the \texttt{shortcuts=ac} or \texttt{shortcuts=acronyms} package option, respectively.

\texttt{\acf\{\langle\text{options}\rangle\}\{\langle\text{entry-label}\rangle\}\{\langle\text{inset}\rangle\} \quad \text{modifiers: } * + \langle\text{alt-mod}\rangle}

§4.3.2; Table 4.1

A synonym for \Glxsxtrfull or \texttt{\acrfull} defined by the \texttt{shortcuts=ac} or \texttt{shortcuts=acronyms} package option, respectively.

\texttt{\ACFP\{\langle\text{options}\rangle\}\{\langle\text{entry-label}\rangle\}\{\langle\text{inset}\rangle\} \quad \text{modifiers: } * + \langle\text{alt-mod}\rangle}

§4.3.2; Table 4.1

A synonym for \Glxsxtrfullpl defined by the \texttt{shortcuts=ac} package option.

\texttt{\Acfp\{\langle\text{options}\rangle\}\{\langle\text{entry-label}\rangle\}\{\langle\text{inset}\rangle\} \quad \text{modifiers: } * + \langle\text{alt-mod}\rangle}

§4.3.2; Table 4.1

A synonym for \Glxsxtrfullpl or \texttt{\acrfullpl} defined by the \texttt{shortcuts=ac} or \texttt{shortcuts=acronyms} package option, respectively.

\texttt{\acfp\{\langle\text{options}\rangle\}\{\langle\text{entry-label}\rangle\}\{\langle\text{inset}\rangle\} \quad \text{modifiers: } * + \langle\text{alt-mod}\rangle}

§4.3.2; Table 4.1

A synonym for \Glxsxtrlong defined by the \texttt{shortcuts=ac} package option.

\texttt{\Acl\{\langle\text{options}\rangle\}\{\langle\text{entry-label}\rangle\}\{\langle\text{inset}\rangle\} \quad \text{modifiers: } * + \langle\text{alt-mod}\rangle}

§4.3.2; Table 4.1
A synonym for \Glsxtrlong or \Acrlong defined by the shortcuts=ac or shortcuts=acronyms package option, respectively.

\acl{⟨options⟩}{⟨entry-label⟩}{⟨inset⟩}  
\textit{modifiers:} * + ⟨alt-mod⟩

§4.3.2; Table 4.1

A synonym for \glsxtrlong or \acrlong defined by the shortcuts=ac or shortcuts=acronyms package option, respectively.

\ACL{⟨options⟩}{⟨entry-label⟩}{⟨inset⟩}  
\textit{modifiers:} * + ⟨alt-mod⟩

§4.3.2; Table 4.1

A synonym for \GLSxtrlongpl defined by the shortcuts=ac package option.

\Aclp{⟨options⟩}{⟨entry-label⟩}{⟨inset⟩}  
\textit{modifiers:} * + ⟨alt-mod⟩

§4.3.2; Table 4.1

A synonym for \Glsxtrlongpl or \Acrlongpl defined by the shortcuts=ac or shortcuts=acronyms package option, respectively.

\aclp{⟨options⟩}{⟨entry-label⟩}{⟨inset⟩}  
\textit{modifiers:} * + ⟨alt-mod⟩

§4.3.2; Table 4.1

A synonym for \cGLSpl defined by the shortcuts=ac package option.

\Acp{⟨options⟩}{⟨entry-label⟩}{⟨inset⟩}  
\textit{modifiers:} * + ⟨alt-mod⟩

§4.3.2; Table 4.1

A synonym for \cGlspl or \glspl defined by the shortcuts=ac or shortcuts=acronyms package option, respectively.
A synonym for `\cglsp1` or `\glsp1` defined by the `shortcuts=ac` or `shortcuts=acronyms` package option, respectively.

\begin{itemize}
\item \texttt{\acp{\langle options\rangle}{\langle entry-label\rangle}{\langle inset\rangle}} \quad \text{modifiers:} \ast + \langle alt-mod\rangle
\end{itemize}

As `\acrfull` but sentence case.

\begin{itemize}
\item \texttt{\acrfull{\langle options\rangle}{\langle entry-label\rangle}{\langle insert\rangle}} \quad \text{modifiers:} \ast + \langle alt-mod\rangle
\end{itemize}

Displays the full form of an acronym. Only for use with the base glossaries package’s acronym mechanism. This command is not compatible with `\newabbreviation`.

\begin{itemize}
\item \texttt{\acrfullpl{\langle options\rangle}{\langle entry-label\rangle}{\langle insert\rangle}} \quad \text{modifiers:} \ast + \langle alt-mod\rangle
\end{itemize}

As `\acrfullpl` but sentence case.

\begin{itemize}
\item \texttt{\acrfullpl{\langle options\rangle}{\langle entry-label\rangle}{\langle insert\rangle}} \quad \text{modifiers:} \ast + \langle alt-mod\rangle
\end{itemize}

Displays the plural full form of an acronym. Only for use with the base glossaries package’s acronym mechanism. This command is not compatible with `\newabbreviation`.

\begin{itemize}
\item \texttt{\acrlong{\langle options\rangle}{\langle entry-label\rangle}{\langle insert\rangle}} \quad \text{modifiers:} \ast + \langle alt-mod\rangle
\end{itemize}

As `\acrlong` but sentence case.

\begin{itemize}
\item \texttt{\acrlong{\langle options\rangle}{\langle entry-label\rangle}{\langle insert\rangle}} \quad \text{modifiers:} \ast + \langle alt-mod\rangle
\end{itemize}

Displays the long form of an acronym. Only for use with the base glossaries package’s acronym mechanism. This command is not compatible with `\newabbreviation`.  

657
Command Summary

\texttt{\textbackslash acrlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩}} \hspace{1cm} \textit{modifiers: * + (alt-mod)}

As \texttt{\textbackslash acrlong} but sentence case.

\texttt{\textbackslash acrlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩}} \hspace{1cm} \textit{modifiers: * + (alt-mod)}

Displays the plural long form of an acronym. Only for use with the base glossaries package’s acronym mechanism. This command is not compatible with \texttt{\textbackslash newabbreviation}.

\texttt{\textbackslash acronymfont{⟨text⟩}} \hspace{1cm} \textit{glossaries v1.0+}

Used to encapsulate the acronym short form on subsequent use by the base glossaries package. This is redefined by glossaries-extra to use \texttt{\glsabbrvfont}.

\texttt{\textbackslash acronymname} \hspace{1cm} \textit{initial: Acronyms glossaries (language-sensitive)}

Expands to the title of the acronym glossary.

\texttt{\textbackslash acronymtype} \hspace{1cm} \textit{initial: \glsdefaulttype glossaries} \hspace{1cm} §4.1.4; 44

Expands to the label of the default acronym glossary. The \texttt{acronym} or \texttt{acronyms} package option will redefine this to \texttt{acronym}. The \texttt{abbreviations} package option will redefine this to \texttt{\glsxtrabbrvtype} if \texttt{acronyms/acronym} isn’t used.

\texttt{\textbackslash acrpluralsuffix}

Used for the plural suffixes for the base package’s acronym mechanism. Not used with glossaries-extra’s abbreviations, which use \texttt{\glsxtrabbrvpluralsuffix}, \texttt{\abbrvplural-suffix} and commands provided for use with particular abbreviation styles. This command should not be used with glossaries-extra.
\acrshort{options}{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

As \acrshort{options}{⟨entry-label⟩}{⟨insert⟩} but sentence case.

\acs{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

Displays the short form of an acronym. Only for use with the base glossaries package’s acronym mechanism. This command is not compatible with \newabbreviation.

\acspl{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

As \acspl{⟨entry-label⟩}{⟨insert⟩} but sentence case.

\acspl{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

Displays the plural short form of an acronym. Only for use with the base glossaries package’s acronym mechanism. This command is not compatible with \newabbreviation.

\Glsxtrshort{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

A synonym for \GLSxtrshort defined by the shortcuts=ac package option.

\Acrshort{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

A synonym for \Glsxtrshort or \Acroshort defined by the shortcuts=ac or shortcuts =acronyms package option, respectively.

\acs{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

A synonym for \Glsxtrshort or \Acroshort defined by the shortcuts=ac or shortcuts =acronyms package option, respectively.
A synonym for \texttt{\textglsxtrshortpl} defined by the \texttt{shortcuts=ac} package option.

A synonym for \texttt{\textglsxtrshortpl} or \texttt{\textacrshortpl} defined by the \texttt{shortcuts=ac} or \texttt{shortcuts=acronyms} package option, respectively.

A synonym for \texttt{\textglsxtrshortpl} or \texttt{\textacrshortpl} defined by the \texttt{shortcuts=ac} or \texttt{shortcuts=acronyms} package option, respectively.

A synonym for \texttt{\textglsxtrfull} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\textglsxtrfull} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\textglsxtrfull} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\textglsxtrfullpl} defined by the \texttt{shortcuts=abbreviations} package option.
A synonym for \texttt{\LaTeX} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\LaTeX} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\LaTeX} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\LaTeX} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\LaTeX} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\LaTeX} defined by the \texttt{shortcuts=abbreviations} package option.

A synonym for \texttt{\LaTeX} defined by the \texttt{shortcuts=abbreviations} package option.
Command Summary

A synonym for \glsxtrlongpl defined by the \texttt{shortcuts=abbreviations} package option.

\texttt{\textbackslash Alpha} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+} \hspace{1cm} \S 11.5.8; 573

Defined with \texttt{\providecommand}, this just does \texttt{\mathrm{A}}.

\texttt{\textbackslash alsoname} \hspace{1cm} \textit{initial: see also} (language-sensitive)

Used as a cross-reference tag (provided by language packages, such as \texttt{babel}).

\texttt{\textbackslash andname} \hspace{1cm} \textit{initial:} \&

Used by \texttt{\glsseelastsep} (provided by \texttt{glossaries} if not already defined).

\texttt{\textbackslash apptoglossarypreamble}\texttt{[\langle type\rangle]}\texttt{\{\langle text\rangle\}} \hspace{1cm} \texttt{glossaries-extra v1.12+} \hspace{1cm} \S 8.2; 377

Appends (locally) \texttt{\langle text\rangle} to the preamble for the glossary identified by \texttt{\langle type\rangle}. If \texttt{\langle type\rangle} is omitted, \texttt{\glsdefaulttype} is assumed.

\texttt{\AS}\texttt{\langle options\rangle}\texttt{\{\langle entry-label\rangle\}}\texttt{\{\langle inset\rangle\}} \hspace{1cm} \textit{modifiers:} \texttt{* + \langle alt-mod\rangle} \hspace{1cm} \S 4.3.2; Table 4.1

A synonym for \texttt{\Glsxtrshort} defined by the \texttt{shortcuts=abbreviations} package option.

\texttt{\As}\texttt{\langle options\rangle}\texttt{\{\langle entry-label\rangle\}}\texttt{\{\langle inset\rangle\}} \hspace{1cm} \textit{modifiers:} \texttt{* + \langle alt-mod\rangle} \hspace{1cm} \S 4.3.2; Table 4.1

A synonym for \texttt{\Glsxtrshort} defined by the \texttt{shortcuts=abbreviations} package option.

\texttt{\as}\texttt{\langle options\rangle}\texttt{\{\langle entry-label\rangle\}}\texttt{\{\langle inset\rangle\}} \hspace{1cm} \textit{modifiers:} \texttt{* + \langle alt-mod\rangle} \hspace{1cm} \S 4.3.2; Table 4.1

A synonym for \texttt{\Glsxtrshort} defined by the \texttt{shortcuts=abbreviations} package option.
A synonym for \GLSxtrshortpl defined by the {"shortcuts=abbreviations"} package option.

A synonym for \Glsxtrshortpl defined by the {"shortcuts=abbreviations"} package option.

A synonym for \Glsxtrshortpl defined by the {"shortcuts=abbreviations"} package option.

\begin{itemize}
\item \Beta
\end{itemize}

Defined with \providecommand, this just does \texttt{\mathrm{B}}.

Indicates that the \texttt{bib2gls} records are in the file identified in the argument \texttt{(filename)}, which corresponds to the file \texttt{(basename).aux} identified in the option \texttt{bibglsaux=(basename)}.

Hook written to the \texttt{glistex} file identifying the dual label prefix.

\bibglsnewdualindexsymbolsecondary{\langle label\rangle}{\langle options\rangle}{\langle name\rangle}{\langle description\rangle}
Command Summary

Defines secondary terms provided with @dualindexsymbol.

\bibglsprefixlabel{⟨label-prefix⟩} \bib2glsv1.8+ §11.5.7; 571

Hook written to the glstex file identifying the primary label prefix.

\bibglslastgrouptitle{⟨cs⟩}{⟨specs⟩} bib2gls

Sets the last group title.

\bibglsetlocationrecordcount{⟨entry-label⟩}{⟨counter⟩}{⟨location⟩}{⟨value⟩} \bib2glsv1.1+ §11.4.2; 532

Sets the location record count for the given entry.

\bibglsetrecordcount{⟨entry-label⟩}{⟨counter⟩}{⟨value⟩} \bib2glsv1.1+ §11.4.2; 532

Sets the ⟨counter⟩ record count for the given entry.

\bibglsettotalrecordcount{⟨entry-label⟩}{⟨value⟩} \bib2glsv1.1+ §11.4.2; 531

Sets the total record count for the given entry.

\bibglstertiaryprefixlabel{⟨label-prefix⟩} \bib2glsv1.8+ §11.5.7; 571

Hook written to the glstex file identifying the tertiary label prefix.

\capitalisefmtwords{⟨text⟩} \mfirstucv2.03+ 664
Command Summary

Converts \langle text \rangle to title case, where \langle text \rangle may contain text-block commands. The starred form only permits a text-block command at the start of the argument. Limitations apply, see the mfirstuc documentation for further details, either:

```
texdoc mfirstuc
```
or visit ctan.org/pkg/mfirstuc.

\[\texttt{\textbackslash\texttt{\textbackslash\texttt{capitalisewords}}\{\langle \texttt{text} \rangle \}}\]
mfirstuc v1.06+

Converts \langle text \rangle to title case. Limitations apply, see the mfirstuc documentation for further details, either:

```
texdoc mfirstuc
```
or visit ctan.org/pkg/mfirstuc.

\[\texttt{\textbackslash\texttt{\textbackslash\texttt{cGLS}}\{\langle \texttt{options} \rangle\}\{\langle \texttt{entry-label} \rangle\}\{\langle \texttt{insert} \rangle\}}\]
\textit{modifiers:} * \textit{+ \langle alt-mod \rangle}
§6.1; 311

Like \texttt{\textbackslash\texttt{GLS}} but hooks into the entry counting mechanism.

\[\texttt{\textbackslash\texttt{cGls}}\{\langle \texttt{options} \rangle\}\{\langle \texttt{entry-label} \rangle\}\{\langle \texttt{insert} \rangle\}}\]
glossaries v4.14+
\textit{modifiers:} * \textit{+ \langle alt-mod \rangle}
§6.1; 310

Like \texttt{\textbackslash\texttt{Gls}} but hooks into the entry counting mechanism.

\[\texttt{\textbackslash\texttt{cgls}}\{\langle \texttt{options} \rangle\}\{\langle \texttt{entry-label} \rangle\}\{\langle \texttt{insert} \rangle\}}\]
glossaries v4.14+
\textit{modifiers:} * \textit{+ \langle alt-mod \rangle}
§6.1; 309

Like \texttt{\textbackslash\texttt{gls}} but hooks into the entry counting mechanism.

\[\texttt{\textbackslash\texttt{\textbackslash\texttt{cGLSformat}}}\{\langle \texttt{entry-label} \rangle\}\{\langle \texttt{insert} \rangle\}}\]
§6.1; 311

665
Command Summary

Format used by \cGLS if the entry was not used more than the given trigger value on the previous run.

\cGlsformat{⟨entry-label⟩}{⟨insert⟩} glossaries v4.14+ §6.1; 310

Format used by \cGls if the entry was not used more than the given trigger value on the previous run.

\cGlsformat{⟨entry⟩}{⟨insert⟩} glossaries v4.14+ §6.1; 310

Format used by \cGls if the entry was not used more than the given trigger value on the previous run.

\cGLSpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * ++ ⟨alt-mod⟩ §6.1; 311

Like \GLSp1 but hooks into the entry counting mechanism.

\cGLSpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * ++ ⟨alt-mod⟩ §6.1; 310

Like \Glspl but hooks into the entry counting mechanism.

\cGlspl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩] modifiers: * ++ ⟨alt-mod⟩ §6.1; 310

Like \Glspl but hooks into the entry counting mechanism.

\cGLSp1format{⟨entry-label⟩}{⟨insert⟩} §6.1; 311

Format used by \cGLSp1 if the entry was not used more than the given trigger value on the previous run.
## Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{\cGlsplformat}(\text{entry-label})\{\text{insert}\}</code></td>
<td>Format used by <code>\cGlspl</code> if the entry was not used more than the given trigger value on the previous run.</td>
<td>§6.1; 311</td>
</tr>
<tr>
<td><code>{\cglspformat}(\text{entry-label})\{\text{insert}\}</code></td>
<td>Format used by <code>\cglsp</code> if the entry was not used more than the given trigger value on the previous run.</td>
<td>§6.1; 310</td>
</tr>
<tr>
<td><code>{\Chig}</code></td>
<td>Defined with <code>\providecommand</code>, this just does <code>\text{mathrm}\{X\}</code>.</td>
<td>§11.5.8; 573</td>
</tr>
<tr>
<td><code>{\csGlsXtrLetField}(\text{entry-label})\{\text{field-label}\}\{\text{cs-name}\}</code></td>
<td>Like <code>\GlsXtrLetField</code> but internally uses (etoolbox’s) <code>\csletcs</code> instead of <code>\cslet</code>.</td>
<td>§3.5; 40</td>
</tr>
<tr>
<td><code>{\currentglossary}</code></td>
<td>Defined by the <code>\print...glossary</code> commands to the current glossary label.</td>
<td>§4.5.3.1; 165</td>
</tr>
<tr>
<td><code>{\CustomAbbreviationFields}</code></td>
<td>Expands to the default field definitions for the entry.</td>
<td></td>
</tr>
<tr>
<td><code>{\defglsentryfmt}[\text{type}]\{\text{display}\}</code></td>
<td>Overrides the default display format (<code>\glsentryfmt</code>) for the given glossary. If <code>\text{type}</code> is omitted, <code>\glsdefaulttype</code> is assumed. This will make the <code>\gls</code>-like commands do <code>\text{display}</code>.</td>
<td>667</td>
</tr>
</tbody>
</table>
for any entries that have the type field set to the given ⟨type⟩. If you want to support any abbreviation styles, you need to include \glssetabbrvfmt in ⟨display⟩. Non-regular abbreviation styles are designed to work with \glsxtrgenabbrvfmt.

\delimN

Used as a separator between locations.

\delimR

Used between the start and end of a location range.

\descriptionname

(language-sensitive)

initial: Description glossaries

Expands to the title of the description column for headed tabular-like styles.

\dGLS{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩}  modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+

As \dgls but uses \GLS.

\dGls{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩}  modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+

As \dgls but uses \Gls.

\dgls{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩}  modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+

Does \gls{⟨options⟩}{⟨prefix⟩⟨entry-label⟩}{⟨insert⟩} for the first prefix in the prefix list that matches a defined entry.
\dGlsdisp[⟨options⟩]{⟨entry-label⟩}{⟨text⟩}  
\[ \text{modifiers: * + ⟨alt-mod⟩} \]
glossaries-extra-bib2gls v1.49+

As \dglssdisp but applies sentence case.

\dGlsdisp[⟨options⟩]{⟨entry-label⟩}{⟨text⟩}  
\[ \text{modifiers: * + ⟨alt-mod⟩} \]
glossaries-extra-bib2gls v1.37+

As \dglss but uses \glsdisp.

\dGLSfield[⟨options⟩]{⟨entry-label⟩}{⟨field-label⟩}{⟨text⟩}  
\[ \text{modifiers: * + ⟨alt-mod⟩} \]
glossaries-extra-bib2gls v1.49+

As \dglssfield but all caps.

\dGLSfield[⟨options⟩]{⟨entry-label⟩}{⟨field-label⟩}{⟨text⟩}  
\[ \text{modifiers: * + ⟨alt-mod⟩} \]
glossaries-extra-bib2gls v1.49+

As \dglssfield but applies sentence case.

\dglssfield[⟨options⟩]{⟨entry-label⟩}{⟨field-label⟩}{⟨text⟩}  
\[ \text{modifiers: * + ⟨alt-mod⟩} \]
glossaries-extra-bib2gls v1.49+

As \dglss but selects the first matching label that has an entry with the field set.

\dglssfieldactualfieldlabel  
glossaries-extra-bib2gls v1.49+

Set by the \dglssfield family of commands to the actual field used. This will either be the requested field or the fallback field.

\dglssfieldcurrentfieldlabel  
glossaries-extra-bib2gls v1.49+
Set by the `\dglasfield` family of commands to the given *⟨field-label⟩*.

```
\dglasfieldfallbackfieldlabel initial: text glossaries-extra-bib2gls v1.49+
```

Expands to the fallback field to use for the `\dglasfield` family of commands.

```
\dGlslink[⟨options⟩]{⟨entry-label⟩}{⟨text⟩} modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.49+
```

As `\dglaslink` but applies sentence case.

```
\dglaslink[⟨options⟩]{⟨entry-label⟩}{⟨text⟩} modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+
```

As `\dglas` but uses `\glslink`.

```
\dGLSpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+
```

As `\dGLSpl` but uses `\GLSpl`.

```
\dGlspl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+
```

As `\dGlspl` but uses `\Glspl`.

```
\dglsp[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+
```

As `\dglsp` but uses `\glsp`.

```
\dglsp[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
glossaries-extra-bib2gls v1.37+
```

As `\dglsp` but uses `\glsp`. 
\Digamma \providecommand \mathrm{F}.

\doifglossarynoexistsordo{⟨glossary-type⟩}{⟨true⟩}{⟨false⟩} \glossaries v4.19+

If the glossary given by ⟨glossary-type⟩ doesn’t exist, this does ⟨true⟩, otherwise it generates an error and does ⟨false⟩. This uses the starred form of \ifglossaryexists to test for existence.

\DTLformatlist{⟨csv-list⟩} \datatool–base v2.28+

Formats the comma-separated list ⟨csv-list⟩. One-level expansion is performed on ⟨csv-list⟩. See the datatool documentation for further details, either:

texdoc datatool

or visit ctan.org/pkg/datatool.

\DTLifinlist{⟨element⟩}{⟨csv-list⟩}{⟨true⟩}{⟨false⟩} \datatool–base

Does ⟨true⟩ if ⟨element⟩ is contained in the comma-separated list ⟨csv-list⟩, otherwise does ⟨false⟩. One-level expansion is performed on ⟨csv-list⟩, but not on ⟨element⟩. See the datatool documentation for further details, either:

texdoc datatool

or visit ctan.org/pkg/datatool.

E

\eglssetwidest[⟨level⟩]{⟨name⟩} \glossaries–extra–stylemods v1.05+

As \glsssetwidest but expands ⟨text⟩.
### Command Summary

- **\eglsupdatewidest**\[\langle level\rangle\langle name\rangle\]  
  - `glossaries-extra-stylemods v1.23+
  - As `\glsupdatewidest` but expands `\langle name\rangle`.

- **\eGlSXtrSetField**\{\langle entry-label\rangle\}\{\langle field-label\rangle\}\{\langle value\rangle\}  
  - `glossaries-extra v1.12+
  - As `\GlSXtrSetField` but expands the value.

- **\entryname**  
  - `initial: Notation ` `glossaries`
  - (language-sensitive)
  - Expands to the title of the name column for headed tabular-like styles.

- **\Epsilon**  
  - `glossaries-extra-bib2gls v1.27+
  - Defined with `\providecommand`, this just does `\mathrm{E}`.

- **\Eta**  
  - `glossaries-extra-bib2gls v1.27+
  - Defined with `\providecommand`, this just does `\mathrm{H}`.

- **\ExtraCustomAbbreviationFields**  
  - `glossaries-extra v1.31+
  - Expands to additional fields that need to be set with `\newabbreviation`.

### F

- **\firstacronymfont**\{\langle text\rangle\}  
  - `glossaries v1.14+
  - Used to encapsulate the acronym short form on first use by the base `glossaries` package. This is redefined by `glossaries-extra` to use `\glsfirstabbrvfont`.

---

672
Command Summary

\texttt{\forallabbreviationlists\{\langle cs \rangle\}\{\langle body \rangle\}} \quad \text{glossaries-extra v1.42+}

Iterates overall all lists of abbreviations, defines the command \langle cs \rangle to the current label and does \langle body \rangle.

\texttt{\forallacronyms\{\langle cs \rangle\}\{\langle body \rangle\}} \quad \text{glossaries v4.08+}

Iterates overall all glossaries that have been declared lists of acronyms, defines the command \langle cs \rangle to the current label and does \langle body \rangle. Use \texttt{\forallabbreviationlists} with glossaries-extra.

\texttt{\forallglossaries[\{\langle types \rangle\}\{\langle cs \rangle\}\{\langle body \rangle\}}} \quad \text{glossaries}

Iterates overall all the glossary labels given in the \langle types \rangle argument, defines the command \langle cs \rangle to the current label and does \langle body \rangle. If the optional argument is omitted, the list of all non-ignored glossaries is assumed.

\texttt{\forallglsentries[\{\langle types \rangle\}\{\langle cs \rangle\}\{\langle body \rangle\}}} \quad \text{glossaries}

Does \texttt{\forallglsentries} for each glossary. The optional argument \langle types \rangle is a comma-separated list of glossary labels. If omitted, all non-ignored glossaries is assumed.

\texttt{\forallglsentries[\{\langle type \rangle\}\{\langle cs \rangle\}\{\langle body \rangle\}}} \quad \text{glossaries}

Iterates over all entries in the given glossary and, at each iteration, defines the command \langle cs \rangle to the current entry label and does \langle body \rangle. The optional argument \langle type \rangle is the glossary label and defaults to \texttt{\glsdefaulttype} if omitted. This command can't be used with \texttt{bib2gls} since there are no defined entries until \texttt{bib2gls} has selected them and added them to the \texttt{glstex} file.

\texttt{\glssetwidest[\{\langle level \rangle\}\{\langle name \rangle\}]} \quad \text{glossaries-extra-stylemods v1.21+}

G

\texttt{\glssetwidest[\{\langle level \rangle\}\{\langle name \rangle\}]} \quad \text{glossaries-extra-stylemods v1.21+}

\texttt{\glssetwidest[\{\langle level \rangle\}\{\langle name \rangle\}]}
Command Summary

As \glsetwidest but global.

\glsupdatewidest\{⟨level⟩\}\{⟨name⟩\}  glossaries-extra-stylemods v1.23+

As \glsupdatewidest but global.

\GlsXtrSetField\{⟨entry-label⟩\}{⟨field-label⟩}{⟨value⟩}  glossaries-extra v1.12+

As \GlsXtrSetField but globally assigns the value.

Glo

\glolinkprefix  \textit{initial: glo:} glossaries

Expands to the default prefix for the entry’s hypertarget anchor in the glossary.

\GlossariesAbbrStyleTooComplexWarning  glossaries-extra v1.49+

Issues a warning with \GlossariesExtraWarning when a command is used that isn’t supported by a complex abbreviation style.

\glossariesextrasetup\{⟨options⟩\}  §2; 8

Change allowed options that are defined or modified by the glossaries-extra package. Note that some options can only be passed as package options.

\GlossariesExtraWarning\{⟨message⟩\}

Writes a warning in the transcript with the current line number. The nowarn option redefines this command to do nothing.
\GlossariesExtraWarningNoLine\{\textlangle message\textrangle\}\glossaries-extra

Writes a warning in the transcript without a corresponding line number. The nowarn option redefines this command to do nothing.

\glossaryentrynumbers\{\textlangle location list\textrangle\}\glossaries

Used within the glossary to encapsulate the location list (redefined by the nonumberlist option).

\glossaryheader\glossaries

Inserted after \begin{theglossary}. This command should be redefined by the glossary style.

\glossaryname initial: Glossary (language-sensitive)\glossaries

Expands to the default glossary title (provided by glossaries if not already defined).

\glossarypostamble\glossaries

Used at the end of the glossary.

\glossarypreamble\glossaries

Used at the start of the glossary. This will be locally redefined to the preamble associated with the current glossary, if one has been set.

\glossarysection\{\textlangle toc-title\textrangle\}{\textlangle title\textrangle}\glossaries

Occurs at the start of a glossary (except with \printunsrtinnerglossary). This will typically be defined to use a sectioning command, such as \section or \chapter. The default definition follows the section and numberedsection options.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glossarytitle</td>
<td>Initialised by the \printglossary set of commands (such as \printglossary) to the current glossary’s title.</td>
</tr>
<tr>
<td>\glossarytoctitle</td>
<td>Initialised by the \printglossary set of commands (such as \printglossary) to the current glossary’s table of contents title.</td>
</tr>
<tr>
<td>\glossentry</td>
<td>Used to format a top-level entry. This command should be redefined by the glossary style.</td>
</tr>
<tr>
<td>\Glossentrydesc</td>
<td>As \glossentrydesc but sentence case.</td>
</tr>
<tr>
<td>\glossentrydesc</td>
<td>Used by glossary styles to display the entry’s description.</td>
</tr>
<tr>
<td>\Glossentryname</td>
<td>As \glossentryname but sentence case.</td>
</tr>
<tr>
<td>\glossentryname</td>
<td>Used by glossary styles to display the entry’s name.</td>
</tr>
<tr>
<td>\glossentrynameother</td>
<td></td>
</tr>
</tbody>
</table>

§8.6; 419

676
Command Summary

Behaves like \glossentryname but uses the given field (identified by its internal label) instead of name.

\Glossentriesymbol\{\langle entry-label \rangle \}
glossaries v3.08a+

As \glossentriesymbol but sentence case.

\glossentriesymbol\{\langle entry-label \rangle \}
glossaries v3.08a+

Used by glossary styles to display the entry’s symbol.

\glossxtrsetpopts
glossaries-extra v1.07+

Used at the start of each glossary to set the current options for the \glsxtrp set of commands (with \glossxtrsetpopts).

\GLS\[\langle options \rangle\]\{\langle entry-label \rangle \}[\langle insert \rangle]
modifiers: * + \langle alt-mod \rangle
glossaries

As \gls but converts the link text to all caps.

\GLS[\langle options \rangle]\{\langle entry-label \rangle \}[\langle insert \rangle]
modifiers: * + \langle alt-mod \rangle
glossaries

As \gls but converts the first character of the link text to uppercase (for the start of a sentence) using \makefirstuc.

\gls\[\langle options \rangle\]\{\langle entry-label \rangle \}[\langle insert \rangle]
modifiers: * + \langle alt-mod \rangle
glossaries

References the entry identified by \langle entry-label \rangle. The text produced depends on whether or not this is the first use and whether or not the regular attribute has been set. The \langle insert \rangle argument may be inserted at the end of the link text or may be inserted at a different point (for
example, after the long form on first use for some abbreviation styles. For the first optional argument, see \glslink options.

\texttt{\glsabbrvdefaultfont\{⟨text⟩\}}

Formatting command for the short form used by the abbreviation styles that don’t apply a font change by default.

\texttt{\glsabbrvemfont\{⟨text⟩\}}
glossaries-extra v1.04+

Short form font used by the “em” abbreviation styles.

\texttt{\glsabbrvfont\{⟨text⟩\}}

Font formatting command for the short form, initialised by the abbreviation style.

\texttt{\glsabbrvhypenfont\{⟨text⟩\}}
glossaries-extra v1.17+

Short form font used by the “hyphen” abbreviation styles.

\texttt{\glsabbrvonlyfont\{⟨text⟩\}}
glossaries-extra v1.17+

Short form font used by the “only” abbreviation styles.

\texttt{\glsabbrvscfont\{⟨text⟩\}}
glossaries-extra v1.17+

Short form font used by the small caps “sc” abbreviation styles.

\texttt{\glsabbrvsconlyfont\{⟨text⟩\}}
glossaries-extra v1.48+

Short form font used by the “sc-only” styles, such as \texttt{long-only-short-sc-only}. 


**Command Summary**

\glsabbrvscuserfont{(text)}  
Short form font used by the small caps “sc-user” abbreviation styles.

\glsabbrvsmfont{(text)}  
Short form font used by the “sm” abbreviation styles.

\glsabbrvuserfont{(text)}  
Short form font used by the “user” abbreviation styles.

\glsabspace{(label)}  
As \glsacspace but includes inner formatting. Unlike \glsacspace, this command is robust.

\GLSaccessdesc{(entry-label)}  
The all caps version of \glsaccessdesc.

\Glsaccessdesc{(entry-label)}  
The sentence case version of \glsaccessdesc.

\glsaccessdesc{(entry-label)}  
If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the description key with the accessibility support enabled for that key (descriptionaccess). If there is no accessibility support, this just uses \glsentrydesc.
\GLSaccessdescplural\langle entry-label \rangle

The all caps version of \glsaccessdescplural.

\Glsaccessdescplural\langle entry-label \rangle

The sentence case version of \glsaccessdescplural.

\glsaccessdescplural\langle entry-label \rangle

If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the descriptionplural key with the accessibility support enabled for that key (descriptionpluralaccess). If there is no accessibility support, this just uses \glssentrydescplural.

\GLSaccessfirst\langle entry-label \rangle

The all caps version of \glsaccessfirst.

\Glsaccessfirst\langle entry-label \rangle

The sentence case version of \glsaccessfirst.

\glsaccessfirst\langle entry-label \rangle

If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the first key with the accessibility support enabled for that key (first-access). If there is no accessibility support, this just uses \glssentryfirst.

\GLSaccessfirstplural\langle entry-label \rangle

The all caps version of \glsaccessfirstplural.
The sentence case version of \texttt{glsaccessfirstplural}.

\texttt{glsaccessfirstplural}\{\textit{entry-label}\}\}

If accessibility support was enabled when glossaries-extra was loaded (\texttt{accsupp}) this will display the value of the \texttt{firstplural} key with the accessibility support enabled for that key (\texttt{firstpluralaccess}). If there is no accessibility support, this just uses \texttt{glsentryfirstplural}.

\texttt{GLSaccessfmtdesc}\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}\}

Similar to \texttt{GLSaccessdesc} but formats the displayed text with \texttt{GLSfmtfield}.

\texttt{Glsaccessfmtdesc}\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}\}

Similar to \texttt{Glsaccessdesc} but formats the displayed text with \texttt{Glsfmtfield}.

\texttt{glsaccessfmtdesc}\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}\}

Similar to \texttt{glsaccessdesc} but formats the displayed text with \texttt{glsfmtfield}.

\texttt{GLSaccessfmtdescplural}\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}\}

Similar to \texttt{GLSaccessdescplural} but formats the displayed text with \texttt{GLSfmtfield}.

\texttt{Glsaccessfmtdescplural}\{\textit{insert}\}\{\textit{cs}\}\{\textit{entry-label}\}\}

Similar to \texttt{Glsaccessdescplural} but formats the displayed text with \texttt{Glsfmtfield}.  

681
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsaccessfmtdescplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \glsaccessdescplural but formats the displayed text with \glsfmtfield.</td>
<td>9.3</td>
</tr>
<tr>
<td>\GLSaccessfmtfirst{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \GLSaccessfirst but formats the displayed text with \GLSfmtfield.</td>
<td>9.3</td>
</tr>
<tr>
<td>\Glsaccessfmtfirst{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \Glsaccessfirst but formats the displayed text with \Glsfmtfield.</td>
<td>9.3</td>
</tr>
<tr>
<td>\glsaccessfmtfirst{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \glsaccessfirst but formats the displayed text with \glsfmtfield.</td>
<td>9.3</td>
</tr>
<tr>
<td>\GLSaccessfmtfirstplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \GLSaccessfirstplural but formats the displayed text with \GLSfmtfield.</td>
<td>9.3</td>
</tr>
<tr>
<td>\Glsaccessfmtfirstplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \Glsaccessfirstplural but formats the displayed text with \Glsfmtfield.</td>
<td>9.3</td>
</tr>
<tr>
<td>\glsaccessfmtfirstplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \glsaccessfirstplural but formats the displayed text with \glsfmtfield.</td>
<td>9.3</td>
</tr>
<tr>
<td>\GLSaccessfmtlong{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}</td>
<td>similar to \GLSaccessfmtlong but formats the displayed text with \GLSfmtfield.</td>
<td>9.3</td>
</tr>
</tbody>
</table>
Command Summary

Similar to \GLSaccesslong but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtlong\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 489

Similar to \Glsaccessfmtlong but formats the displayed text with \GLSfmtfield.

\glsaccessfmtlong\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 489

Similar to \glsaccessfmtlong but formats the displayed text with \glsfmtfield.

\GLSaccessfmtlongpl\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 489

Similar to \GLSaccessfmtlongpl but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtlongpl\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 489

Similar to \Glsaccessfmtlongpl but formats the displayed text with \glsfmtfield.

\glsaccessfmtlongpl\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 489

Similar to \glsaccessfmtlongpl but formats the displayed text with \glsfmtfield.

\GLSaccessfmtname\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 485

Similar to \GLSaccessfmtname but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtname\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 485

Similar to \Glsaccessfmtname but formats the displayed text with \Glsfmtfield.

\glsaccessfmtname\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

§9.3; 485

Similar to \Glsaccessfmtname but formats the displayed text with \Glsfmtfield.
Command Summary

\glsaccessfmtname\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \glsaccessname but formats the displayed text with \glsfmtfield.

\GLSaccessfmtplural\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \GLSaccessplural but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtplural\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \Glsaccessplural but formats the displayed text with \Glsfmtfield.

\glsaccessfmtplural\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \glsaccessplural but formats the displayed text with \glsfmtfield.

\GLSaccessfmtshort\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \GLSaccessshort but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtshort\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \Glsaccessshort but formats the displayed text with \Glsfmtfield.

\glsaccessfmtshort\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \glsaccessshort but formats the displayed text with \glsfmtfield.

\GLSaccessfmtshortpl\langle insert\rangle\langle cs\rangle\langle entry-label\rangle\glossaries-extra v1.49+

Similar to \GLSaccessshortpl but formats the displayed text with \GLSfmtfield.
**Command Summary**

Similar to \GLSaccessshortpl but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtshortpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}
glossaries-extra v1.49+ §9.3; 488

Similar to \Glsaccessshortpl but formats the displayed text with \Glsfmtfield.

\glsaccessfmtshortpl{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩}
glossaries-extra v1.49+ §9.3; 488

Similar to \glsaccessshortpl but formats the displayed text with \glsfmtfield.

\GLSaccessfmtsymbol{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩} glossaries-extra v1.49+ §9.3; 487

Similar to \GLSaccesssymbol but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtsymbol{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩} glossaries-extra v1.49+ §9.3; 486

Similar to \GLSaccesssymbol but formats the displayed text with \Glsfmtfield.

\glsaccessfmtsymbol{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩} glossaries-extra v1.49+ §9.3; 486

Similar to \glsaccesssymbol but formats the displayed text with \glsfmtfield.

\GLSaccessfmtsymbolplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩} glossaries-extra v1.49+ §9.3; 487

Similar to \GLSaccesssymbolplural but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtsymbolplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩} glossaries-extra v1.49+ §9.3; 487

Similar to \glsaccesssymbolplural but formats the displayed text with \glsfmtfield.

\glsaccessfmtsymbolplural{⟨insert⟩}{⟨cs⟩}{⟨entry-label⟩} glossaries-extra v1.49+ §9.3; 487
Command Summary

\glsaccessfmtsymbolplural\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \glsaccesssymbolplural but formats the displayed text with \glsfmtfield.

\GLSaccessfmttext\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \GLSaccesstext but formats the displayed text with \GLSfmtfield.

\Glsaccessfmttext\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \Glsaccesstext but formats the displayed text with \Glsfmtfield.

\glsaccessfmttext\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \glsaccesstext but formats the displayed text with \glsfmtfield.

\GLSaccessfmtuseri\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \GLSaccessuseri but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtuseri\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \Glsaccessuseri but formats the displayed text with \Glsfmtfield.

\glsaccessfmtuseri\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \glsaccessuseri but formats the displayed text with \glsfmtfield.

\GLSaccessfmtuserii\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}  ~ glossaries-extra v1.49+

Similar to \GLSaccessuserii but formats the displayed text with \GLSfmtfield.
Similar to \GLSaccessuserii but formats the displayed text with \GLSfmtfield.

\GLSaccessfmtuserii\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 490

Similar to \glsaccessuserii but formats the displayed text with \glsfmtfield.

\glsaccessfmtuserii\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 490

Similar to \glsaccessuserii but formats the displayed text with \glsfmtfield.

\glsaccessfmtuserii\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 490

Similar to \GLSaccessuseriii but formats the displayed text with \GLSfmtfield.

\GLSaccessfmtuseriii\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 490

Similar to \GLSaccessuseriii but formats the displayed text with \GLSfmtfield.

\GLSaccessfmtuseriii\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 490

Similar to \GLSaccessuseriii but formats the displayed text with \GLSfmtfield.

\GLSaccessfmtuseriv\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 491

Similar to \GLSaccessuseriv but formats the displayed text with \GLSfmtfield.

\GLSaccessfmtuseriv\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 491

Similar to \GLSaccessuseriv but formats the displayed text with \GLSfmtfield.

\GLSaccessfmtuseriv\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 491

Similar to \GLSaccessuseriv but formats the displayed text with \GLSfmtfield.

\GLSaccessfmtuseriv\{\langle insert\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}
glossaries-extra v1.49+

§9.3; 491

687
\glsaccessfmtuserv\{\langle\text{insert}\rangle}\{\langle cs\rangle}\{\langle entry-label\rangle\}

Similar to \glsaccessuseriv but formats the displayed text with \glsfmtfield.

\GLSaccessfmtuserv\{\langle\text{insert}\rangle}\{\langle cs\rangle}\{\langle entry-label\rangle\}

Similar to \GLSaccessuserv but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtuserv\{\langle\text{insert}\rangle}\{\langle cs\rangle}\{\langle entry-label\rangle\}

Similar to \Glsaccessuserv but formats the displayed text with \Glsfmtfield.

\glsaccessfmtuserv\{\langle\text{insert}\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

Similar to \glsaccessuserv but formats the displayed text with \glsfmtfield.

\GLSaccessfmtuservi\{\langle\text{insert}\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

Similar to \GLSaccessuservi but formats the displayed text with \GLSfmtfield.

\Glsaccessfmtuservi\{\langle\text{insert}\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

Similar to \Glsaccessuservi but formats the displayed text with \Glsfmtfield.

\glsaccessfmtuservi\{\langle\text{insert}\rangle\}\{\langle cs\rangle\}\{\langle entry-label\rangle\}

Similar to \glsaccessuservi but formats the displayed text with \glsfmtfield.

\glsaccessibility[\langle options\rangle]\{\langle PDF element\rangle\}\{\langle value\rangle\}\{\langle content\rangle\}

Similar to \glsaccessuserv but formats the displayed text with \glsfmtfield.
Command Summary

Applies \langle value \rangle as the accessibility attribute \langle PDF element \rangle for the given \langle content \rangle. This internally uses the accessibility support provided by accsupp.

\GLSaccesslong\{\langle entry-label \rangle\}

§9.2; 481

The all caps version of \glsaccesslong.

\Glsaccesslong\{\langle entry-label \rangle\}

§9.2; 481

The sentence case version of \glsaccesslong.

\glsaccesslong\{\langle entry-label \rangle\}

§9.2; 481

If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the long key with the accessibility support enabled for that key (long-access). If there is no accessibility support, this just uses \glsentrylong.

\GLSaccesslongpl\{\langle entry-label \rangle\}

§9.2; 482

The all caps version of \glsaccesslongpl.

\Glsaccesslongpl\{\langle entry-label \rangle\}

§9.2; 482

The sentence case version of \glsaccesslongpl.

\glsaccesslongpl\{\langle entry-label \rangle\}

§9.2; 482

If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the longplural key with the accessibility support enabled for that key (longplural-access). If there is no accessibility support, this just uses \glsentrylongpl.
The all caps version of \glsaccessname.

\GLSaccessname{⟨entry-label⟩}

§9.2; 477

The sentence case version of \glsaccessname.

\Glsaccessname{⟨entry-label⟩}

§9.2; 477

If accessibility support was enabled when glossaries–extra was loaded (\accsupp) this will display the value of the name key with the accessibility support enabled for that key (\access). If there is no accessibility support, this just uses \glsentryname.

\GLSaccessplural{⟨entry-label⟩}

§9.2; 478

The all caps version of \glsaccessplural.

\Glsaccessplural{⟨entry-label⟩}

§9.2; 478

The sentence case version of \glsaccessplural.

\glsaccessplural{⟨entry-label⟩}

§9.2; 478

If accessibility support was enabled when glossaries–extra was loaded (\accsupp) this will display the value of the plural key with the accessibility support enabled for that key (plural-access). If there is no accessibility support, this just uses \glsentryplural.

\GLSaccessshort{⟨entry-label⟩}

§9.2; 481

The all caps version of \glsaccessshort.
\Glsaccessshort\{\langle entry-label \rangle \}\n
The sentence case version of \glsaccessshort.

\glsaccessshort\{\langle entry-label \rangle \}\n
If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the short key with the accessibility support enabled for that key (short-access). If there is no accessibility support, this just uses \glsentryshort.

\GLSaccessshortpl\{\langle entry-label \rangle \}\n
The all caps version of \glsaccessshortpl.

\Glsaccessshortpl\{\langle entry-label \rangle \}\n
The sentence case version of \glsaccessshortpl.

\glsaccessshortpl\{\langle entry-label \rangle \}\n
If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the shortplural key with the accessibility support enabled for that key (shortpluralaccess). If there is no accessibility support, this just uses \glsentryshortpl.

\GLSaccesssymbol\{\langle entry-label \rangle \}\n
The all caps version of \glsaccesssymbol.

\Glsaccesssymbol\{\langle entry-label \rangle \}\n
The sentence case version of \glsaccesssymbol.
If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the symbol key with the accessibility support enabled for that key (symbol-access). If there is no accessibility support, this just uses \glsentriesymbol.

\GLSaccesssymbolplural{⟨entry-label⟩}

The all caps version of \glsaccesssymbolplural.

\Glsaccesssymbolplural{⟨entry-label⟩}

The sentence case version of \glsaccesssymbolplural.

\glsaccesssymbolplural{⟨entry-label⟩}

If accessibility support was enabled when glossaries-extra was loaded (accsupp) this will display the value of the symbolplural key with the accessibility support enabled for that key (symbolplural-access). If there is no accessibility support, this just uses \glsentriesymbolplural.

\GLSaccessplural{⟨entry-label⟩}

The all caps version of \glsaccessplural.

\Glsaccessplural{⟨entry-label⟩}

The sentence case version of \glsaccessplural.
Command Summary

If accessibility support was enabled when glossaries-extra was loaded (`accsupp`) this will display the value of the `text` key with the accessibility support enabled for that key (`text-access`). If there is no accessibility support, this just uses \glsentrytext.

\GLSaccessuseri\{⟨entry-label⟩\}  
\texttt{glossaries-extra v1.49+}  
§9.2; 482

The all caps version of \glsaccessuseri.

\Glsaccessuseri\{⟨entry-label⟩\}  
\texttt{glossaries-extra v1.49+}  
§9.2; 482

The sentence case version of \glsaccessuseri.

\glaccessuseri\{⟨entry-label⟩\}  
\texttt{glossaries-extra v1.49+}  
§9.2; 482

If accessibility support was enabled when glossaries-extra was loaded (`accsupp`) this will display the value of the `user1` key with the accessibility support enabled for that key (`user1-access`). If there is no accessibility support, this just uses \glsentryuseri.

\GLSaccessuserii\{⟨entry-label⟩\}  
\texttt{glossaries-extra v1.49+}  
§9.2; 483

The all caps version of \glsaccessuserii.

\Glsaccessuserii\{⟨entry-label⟩\}  
\texttt{glossaries-extra v1.49+}  
§9.2; 482

The sentence case version of \glsaccessuserii.

\glaccessuserii\{⟨entry-label⟩\}  
\texttt{glossaries-extra v1.49+}  
§9.2; 482

If accessibility support was enabled when glossaries-extra was loaded (`accsupp`) this will display the value of the `user2` key with the accessibility support enabled for that key (`user2-access`). If there is no accessibility support, this just uses \glsentryuserii.
\GLSaccessuseriii\{⟨entry-label⟩}\} \quad \text{glossaries-extra v1.49+} \quad \S9.2; 483

The all caps version of \glsaccessuseriii.

\Glsaccessuseriii\{⟨entry-label⟩}\} \quad \text{glossaries-extra v1.49+} \quad \S9.2; 483

The sentence case version of \glsaccessuseriii.

\glsaccessuseriii\{⟨entry-label⟩\} \quad \text{glossaries-extra v1.49+} \quad \S9.2; 483

If accessibility support was enabled when glossaries-extra was loaded (\acs{accsupp}) this will display the value of the \texttt{user3} key with the accessibility support enabled for that key (\texttt{user3-access}). If there is no accessibility support, this just uses \glsentryuseriii.

\GLSaccessuseriv\{⟨entry-label⟩\} \quad \text{glossaries-extra v1.49+} \quad \S9.2; 483

The all caps version of \glsaccessuseriv.

\Glsaccessuseriv\{⟨entry-label⟩\} \quad \text{glossaries-extra v1.49+} \quad \S9.2; 483

The sentence case version of \glsaccessuseriv.

\glsaccessuseriv\{⟨entry-label⟩\} \quad \text{glossaries-extra v1.49+} \quad \S9.2; 483

If accessibility support was enabled when glossaries-extra was loaded (\acs{accsupp}) this will display the value of the \texttt{user4} key with the accessibility support enabled for that key (\texttt{user4-access}). If there is no accessibility support, this just uses \glsentryuseriv.

\GLSaccessuserv\{⟨entry-label⟩\} \quad \text{glossaries-extra v1.49+} \quad \S9.2; 484

The all caps version of \glsaccessuserv.
\texttt{\textbackslash Glsaccessuserv\{\langle entry-label\rangle\}} \hspace{1em} \texttt{glossaries-extra v1.49+} \hspace{1em} \S9.2; 484

The sentence case version of \texttt{\textbackslash Glsaccessuserv}.

\texttt{\textbackslash glsaccessuserv\{\langle entry-label\rangle\}} \hspace{1em} \texttt{glossaries-extra v1.49+} \hspace{1em} \S9.2; 483

If accessibility support was enabled when \texttt{glossaries-extra} was loaded (\texttt{accsupp}) this will display the value of the \texttt{user5} key with the accessibility support enabled for that key (\texttt{user5-access}). If there is no accessibility support, this just uses \texttt{\textbackslash glsentryuserv}.

\texttt{\textbackslash GLSaccessuservi\{\langle entry-label\rangle\}} \hspace{1em} \texttt{glossaries-extra v1.49+} \hspace{1em} \S9.2; 484

The all caps version of \texttt{\textbackslash GLSaccessuservi}.

\texttt{\textbackslash Glsaccessuservi\{\langle entry-label\rangle\}} \hspace{1em} \texttt{glossaries-extra v1.49+} \hspace{1em} \S9.2; 484

The sentence case version of \texttt{\textbackslash Glsaccessuservi}.

\texttt{\textbackslash glsaccessuservi\{\langle entry-label\rangle\}} \hspace{1em} \texttt{glossaries-extra v1.49+} \hspace{1em} \S9.2; 484

If accessibility support was enabled when \texttt{glossaries-extra} was loaded (\texttt{accsupp}) this will display the value of the \texttt{user6} key with the accessibility support enabled for that key (\texttt{user6-access}). If there is no accessibility support, this just uses \texttt{\textbackslash glsentryuservi}.

\texttt{\textbackslash glsaccsupp\{\langle replacement\rangle\}\{\langle content\rangle\}} \hspace{1em} \texttt{glossaries-accsupp} \hspace{1em} \S9.2; 484

Applies \texttt{\langle replacement\rangle} as the ActualText for \texttt{\langle content\rangle} using \texttt{\textbackslash glsaccessibility}.

\texttt{\textbackslash glsacspace\{\langle label\rangle\}} \hspace{1em} \texttt{glossaries v4.16+} \hspace{1em} \S9.2; 484

Uses a non-breakable space if the short form is less than \texttt{\textbackslash glsacspacemax} otherwise uses \texttt{\space}. This command is provided by \texttt{glossaries} but has a hard-coded maximum of 3em. This command is redefined by \texttt{glossaries-extra} to use \texttt{\textbackslash glsacspacemax}.
**Command Summary**

\texttt{\glssacspacemax}

Expands to the maximum value used by \glssacspace. This is a macro not a register. The default is 3em.

\texttt{\glssadd[⟨options⟩]{⟨entry-label⟩}}

Indexes the entry identified by ⟨entry-label⟩.

\texttt{\glssaddall[⟨options⟩]}

Iterates over all glossaries (or all those listed in the \texttt{types} option) and indexes each entry in the glossary. The optional argument ⟨options⟩ are passed to \texttt{\glssadd}. This command can’t be used with \texttt{bib2gls}. Use the \texttt{selection=all} resource option instead.

\texttt{\glssaddallunindexed[⟨glossary types⟩]}

Iterates over all glossaries listed in ⟨glossary types⟩ and indexes each entry (with \texttt{format=glsignore}) that hasn’t already been indexed. This command can’t be used with \texttt{bib2gls}. Use the \texttt{selection=all} resource option instead.

\texttt{\glssaddallunused[⟨glossary types⟩]}

Iterates over all glossaries listed in ⟨glossary types⟩ and indexes each entry (with \texttt{format=glsignore}) that hasn’t been used. This command can’t be used with \texttt{bib2gls}. Use the \texttt{selection=all} resource option instead.

\texttt{\glssaddeach[⟨options⟩]{⟨entry label list⟩}}

Does \texttt{\glssadd[⟨options⟩]{⟨entry-label⟩}} for each label in the supplied comma-separated list.

\texttt{\glssaddkey{⟨key⟩}{⟨default value⟩}{⟨no link cs⟩}{⟨no link ucfirst cs⟩}{⟨link cs⟩}{⟨link ucfirst cs⟩}{⟨link allcaps cs⟩}}

Iterates over all glossaries listed in ⟨glossary types⟩ and indexes each entry (with \texttt{format=glsignore}) that hasn’t already been indexed. This command can’t be used with \texttt{bib2gls}. Use the \texttt{selection=all} resource option instead.

\texttt{\glssaddeach[⟨options⟩]{⟨entry label list⟩}}

Does \texttt{\glssadd[⟨options⟩]{⟨entry-label⟩}} for each label in the supplied comma-separated list.
Command Summary

Defines a new glossary entry key with the given default value and commands that are analogous to \glsentrytext ({\no link cs}), \Glsentrytext ({\no link ucfirst cs}), \glstext ({\link cs}), \Glstext ({\link ucfirst cs}), \GLStext ({\link allcaps cs}). The starred version switches on field expansion for the given key.

\glsaddpostsetkeys glossaries-extra v1.30+ §5.1.1; 188

Hook implemented after setting the options passed to \glsadd.

\glsaddpresetkeys glossaries-extra v1.30+ §5.1.1; 188

Hook implemented before setting the options passed to \glsadd.

\glsaddstoragekey {\langle key \rangle} {\langle default value \rangle} {\langle no link cs \rangle} glossaries v4.16

Provides a new glossary entry key with a default value and a command for simply accessing the value (without indexing or hyperlinks). The starred version switches on field expansion for the given key.

\glsaltlistitem {\langle entry-label \rangle} glossaries-extra-stylemods v1.47+ §8.6.5.3; 429

Used to display the top-level entry item in the altlist styles.

\glsalttreecchildpredesc glossaries-extra-stylemods v1.46+ §8.6.5.4; 431

Inserted before the child descriptions for the alttree styles.

\glsalttreepredesc glossaries-extra-stylemods v1.46+ §8.6.5.4; 431

Inserted before the top-level descriptions for the alttree styles.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsapptopostlink{\langle category\rangle}{\langle code\rangle}</td>
<td>Appends \langle code\rangle to post-link hook associated with the category identified by the label \langle category\rangle (or simply defines it, if it doesn’t already exist).</td>
</tr>
<tr>
<td>\glsautoprefix</td>
<td>Expands to the prefix for the label used by numberedsection=autolabel and numberedsection=nameref.</td>
</tr>
<tr>
<td>\glsbackslash</td>
<td>Expands to a literal backslash.</td>
</tr>
<tr>
<td>\glscapitalisewords{\langle content\rangle}</td>
<td>Just does \capitalisewords but may be redefined to use \capitalisefmtwords, if required.</td>
</tr>
<tr>
<td>\glscapscase{\langle no change\rangle}{\langle sentence\rangle}{\langle all caps\rangle}</td>
<td>Initialised by the \gls-like and \glstext-like commands, this expands to \langle no change\rangle if the calling command doesn’t apply a case-change (such as \gls or \glstext), to \langle sentence\rangle if the calling command converts to sentence case (such as \Gls or \Glstext), or to \langle all caps\rangle if the calling command converts to all caps (such as \GLS or \GLStext). This command may be used within associated hooks, entry display styles (\defglsentryfmt), and the post-link hook.</td>
</tr>
<tr>
<td>\glscapturedgroup{n}</td>
<td>Expands to \string${n}.</td>
</tr>
</tbody>
</table>

698
Command Summary

\texttt{\textbackslash glscategory\{\langle entry-label\rangle\}}

Expands to the entry’s category.

\texttt{\textbackslash glscategorylabel}

Expands to the category label of the abbreviation that is in the process of being defined by \texttt{\textbackslash newabbreviation}. Maybe used in the style hooks (but take care to expand this command, if necessary).

\texttt{\textbackslash glscombinedfirstsep\{\langle prev label\rangle\}\{\langle next label\rangle\}}

Separator used between elements of a multi-entry set where only the next element have been marked as used.

\texttt{\textbackslash glscombinedfirstsepfirst\{\langle prev label\rangle\}\{\langle next label\rangle\}}

Separator used between elements of a multi-entry set where neither the previous nor the next element has been marked as used.

\texttt{\textbackslash glscombinedsep\{\langle prev label\rangle\}\{\langle next label\rangle\}}

Separator used between elements of a multi-entry set where both elements have been marked as used.

\texttt{\textbackslash glscombinedsepxfirst\{\langle prev label\rangle\}\{\langle next label\rangle\}}

Separator used between elements of a multi-entry set where only the previous element have been marked as used.

\texttt{\textbackslash glscurrententrylabel}

glossaries v3.02+
Assigned at the start of each entry item within the glossary. This command may be used by
glossary hooks, such as the post-description hook, to reference the current entry.

\glscurrententrylevel\hspace{1cm} glossaries-extra v1.44+

Defined within the “unsr” family of commands to the current hierarchical level (taking
leveloffset into account).

\glscurrentfieldvalue\hspace{1cm} glossary v4.23+

Conditional commands such as \ifglshasfield set this to the field’s value for use within
the \langle true\rangle code.

\glscurrentrootentry\hspace{1cm} glossaries-extra v1.49+

May be used within \printunsrtglossaryentryprocesshook to reference the most recent
top level entry label (allowing for flatten but not leveloffset).

\glscurrenttoplevelentry\hspace{1cm} glossaries-extra v1.49+

May be used within \printunsrtglossaryentryprocesshook to reference the most recent
top level entry label (allowing for flatten and leveloffset).

\glscustomtext\hspace{1cm} glossary

The custom text provided by \glsdisp or the link text for the \glstext-like commands.
This command may be used within associated hooks, entry display styles (\defglsentry-
fmt), and the post-link hook.

\glsdefaultshortaccess{⟨long⟩}{⟨short⟩}\hspace{1cm} glossaries-accsupp v4.45+
(requires accsupp)
Command Summary

Used when \newabbreviation automatically assigns shortaccess. This is defined by glossaries–accsupp to just do ⟨long⟩ but is redefined by glossaries–extra to do ⟨long⟩ ⟨⟨short⟩⟩.

\glsdefaulttype
  \textit{initial}: main glossaries

Expands to the label of the default glossary, which is normally main but if \texttt{nomain} is used, it will be the label of the first glossary to be defined.

\glsdefpostdesc\{⟨category⟩\}⟨⟨definition⟩⟩
  \textit{glossaries-extra v1.31+}

§8.6.2; 422

Defines post-description hook associated with the category identified by the label ⟨category⟩. This simply (re)defines \glsxtrpostdesc ⟨category⟩ for the given ⟨category⟩ to ⟨definition⟩.

\glsdefpostlink\{⟨category⟩\}⟨⟨definition⟩⟩
  \textit{glossaries-extra v1.31+}

§5.5.4; 250

Defines post-link hook associated with the category identified by the label ⟨category⟩. This simply (re)defines \glsxtrpostlink ⟨category⟩ for the given ⟨category⟩ to ⟨definition⟩.

\glsdefpostname\{⟨category⟩\}⟨⟨definition⟩⟩
  \textit{glossaries-extra v1.31+}

§8.6.1; 420

Defines post-name hook associated with the category identified by the label ⟨category⟩. This simply (re)defines \glsxtrpostname ⟨category⟩ for the given ⟨category⟩ to ⟨definition⟩.

\GLSdesc\[⟨options⟩\]⟨⟨entry-label⟩⟩[⟨insert⟩]
  \textit{modifiers: * + ⟨alt-mod⟩}

\textit{glossaries}

As \glsdesc but converts the link text to all caps.

\Glsdesc\[⟨options⟩\]⟨⟨entry-label⟩⟩[⟨insert⟩]
  \textit{modifiers: * + ⟨alt-mod⟩}

\textit{glossaries}

As \glsdesc but converts the link text to sentence case. Use \texttt{Glossentrydesc} within custom glossary styles instead of this command.
Command Summary

\glsdesc\[\langle\text{options}\rangle\]{\langle\text{entry-label}\rangle}\{\langle\text{insert}\rangle\}  \textit{modifiers:} * + \langle\text{alt-mod}\rangle  \textit{glossaries}

References the entry identified by \langle\text{entry-label}\rangle. The text produced is obtained from the description value. The \langle\text{insert}\rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \glslink\ options. Use \glossentrydesc\ within custom glossary styles instead of this command.

\glsdescplural\[\langle\text{options}\rangle\]{\langle\text{entry-label}\rangle}\{\langle\text{insert}\rangle\}  \textit{modifiers:} * + \langle\text{alt-mod}\rangle  \textit{glossaries}

As \glsdesc\ but for the descriptionplural field.

\glsdescriptionaccessdisplay\{\langle\text{text}\rangle\}{\langle\text{entry-label}\rangle}  \textit{glossaries-accsupp}

Does \langle\text{text}\rangle with the descriptionaccess replacement text (if set).

\glsdescriptionpluralaccessdisplay\{\langle\text{text}\rangle\}{\langle\text{entry-label}\rangle}  \textit{glossaries-accsupp}

Does \langle\text{text}\rangle with the descriptionpluralaccess replacement text (if set).

\glsdescwidth  \textit{glossary-long & glossary-super}

A length register used to set the width of the description column for tabular-like styles.

\glssdisp\[\langle\text{options}\rangle\]{\langle\text{entry-label}\rangle}\{\langle\text{text}\rangle\}  \textit{modifers:} * + \langle\text{alt-mod}\rangle  \textit{glossaries v4.50+}

As \glssdisp\ but sets the link text to \glssentencecase\{\langle\text{text}\rangle\}. This is provided to allow a sentence case mapping in the event that \glssdisp occurs at the start of content that has automated case-changing.
Command Summary

\glsdisp{(options)}{(entry-label)}{(text)}

modifers: * + {alt-mod}

References the entry identified by \langle entry-label \rangle with the given \langle text \rangle as the link text. This command unsets the first use flag (use \glslink instead, if the first use flag should not be altered). This command is considered a \gls-like command. For the first optional argument, see \glslink options.

\glsdisplaynumberlist{(entry-label)}

Formats the location list for the given entry. Redefined by glossaries-extra-bib2gls to obtain the location list from the location field.

\glsdoifexists{(entry-label)}{(code)}

Does \langle code \rangle if the entry given by \langle entry-label \rangle exists. If the entry doesn’t exist, this will either generate an error (undefaction=error) or a warning (undefaction=warn) and, within the document environment, it will insert the unknown marker ??.

\glsdoifexistsordo{(entry-label)}{(true)}{(false)}

Similar to \ifglsentryexists, this does \langle true \rangle if the entry given by \langle entry-label \rangle exists. If the entry doesn’t exist, this does \langle false \rangle and generates an error (undefaction=error) or a warning (undefaction=warn). The unknown marker ?? will be placed before the \langle false \rangle code.

\glsdoifexistsorwarn{(entry-label)}{(code)}

Like \glsdoifexists, but always warns (no error) if the entry doesn’t exist, regardless of the undefaction setting, and doesn’t show the unknown marker.

\glsdoifnoexists{(entry-label)}{(code)}

703
Does \textit{\langle code\rangle} if the entry given by \textit{\langle entry-label\rangle} does not exist. If the entry does exist, this will either generate an error (\texttt{undefaction=error}) or a warning (\texttt{undefaction=warn}).

\begin{verbatim}
\glSENABLEENTRYCOUNT \glsenableentrycount glossaries v4.14+
\end{verbatim}

Enables entry counting.

\begin{verbatim}
\glSENABLEENTRYUNITCOUNT \glsenableentryunitcount §6.1; 313
\end{verbatim}

Enables entry unit counting.

\begin{verbatim}
\glsencapwrcontent{\langle code\rangle} \glsencapwrcontent glossaries v4.50+ & glossaries-extra v1.49+
\end{verbatim}

Encapsulates the indexing code (within \texttt{\glslinkwrcontent}).

\begin{verbatim}
\glSENTRYCOUNTERLABEL \glsentrycounterlabel glossaries v3.0+
\end{verbatim}

Used by \texttt{\glsentryitem} to display the entry counter label.

\begin{verbatim}
\glsentrycurrcount{\langle entry-label\rangle} \glsentrycurrcount glossaries v4.14+
\end{verbatim}

Expands to the current entry count running total or 0 if not available (needs to be enabled with \texttt{\glsenableentrycount} or \texttt{\glsenableentryunitcount}). With unit entry counting, this expands to the total for the current unit.

\begin{verbatim}
\glsentrydesc{\langle entry-label\rangle} \glsentrydesc glossaries
\end{verbatim}

Partially robust command that displays the value of the \texttt{description} field with sentence case applied. As from \texttt{glossaries v4.50}, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.
**Command Summary**

\glsentrydesc{⟨entry-label⟩}
\glsentrydescplural{⟨entry-label⟩}
\Glsentryfirst{⟨entry-label⟩}
\Glsentryfirstplural{⟨entry-label⟩}

Simply expands to the value of the \texttt{description} field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the \texttt{description} field doesn’t contain any fragile commands.

Simply expands to the value of the \texttt{descriptionplural} field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the \texttt{description-plural} field doesn’t contain any fragile commands.

Partially robust command that displays the value of the \texttt{first} field with the first letter converted to uppercase. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

Partially robust command that displays the value of the \texttt{firstplural} field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.
Simply expands to the value of the `firstplural` field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the `firstplural` field doesn’t contain any fragile commands.

\texttt{\textbackslash glsentryfmt} \textit{glossaries v3.11a+}

The default display format used by the \texttt{\gls} like commands. This checks if the `short` field has been set for the current entry and, if set, initialises the abbreviation formatting commands (with \texttt{\glssetabbrvfmt}). This command will do \texttt{\glsgenentryfmt} (encapsulated with \texttt{\glsxtrregularfont}) if the entry is considered a regular entry (\texttt{\glsifregular}) or if the entry doesn’t have the `short` field set. Otherwise it will do \texttt{\glsxtrgenabbrvfmt} encapsulated with \texttt{\glsxtrabbreviationfont}.

\texttt{\textbackslash glsentryindexcount\{entry-label\}} \textit{glossaries-extra v1.49+} §5.8; 264

Expands to the number of times the given entry has been indexed. This will expand to 0 if the entry hasn’t been indexed or hasn’t been defined.

\texttt{\textbackslash glsentryitem\{entry-label\}} \textit{glossaries v3.0+}

Does nothing if \texttt{entrycounter=false}, otherwise increments and displays the associated counter.

\texttt{\textbackslash Glsentrylong\{entry-label\}} \textit{glossaries v3.0+}

Displays the value of the `long` field with sentence case applied. Does nothing if the entry hasn’t been defined. As from \texttt{glossaries v4.50}, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\texttt{\textbackslash glsentrylong\{entry-label\}} \textit{glossaries v3.0+}

Simply expands to the value of the `long` field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the `long` field doesn’t contain any fragile commands.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\Glsentrylongpl{⟨entry-label⟩}</td>
<td>Displays the value of the <code>longplural</code> field with sentence case applied. Does nothing if the entry hasn’t been defined. As from <code>glossaries</code> v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.</td>
</tr>
<tr>
<td>\glsentrylongpl{⟨entry-label⟩}</td>
<td>Simply expands to the value of the <code>longplural</code> field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the <code>longplural</code> field doesn’t contain any fragile commands.</td>
</tr>
<tr>
<td>\Glsentryname{⟨entry-label⟩}</td>
<td>Partially robust command that displays the value of the <code>name</code> field with sentence case applied. As from <code>glossaries</code> v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.</td>
</tr>
<tr>
<td>\glsentryname{⟨entry-label⟩}</td>
<td>Simply expands to the value of the <code>name</code> key. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the <code>name</code> key doesn’t contain any fragile commands.</td>
</tr>
<tr>
<td>\glsentrynumberlist{⟨entry-label⟩}</td>
<td>Displays the location list for the given entry. Redefined by <code>glossaries-extra-bib2gls</code> to obtain the location list from the <code>location</code> field.</td>
</tr>
<tr>
<td>\glsentryparent{⟨entry-label⟩}</td>
<td>Expands to the value of the <code>parent</code> field. Expands to nothing if the <code>parent</code> field hasn’t been set and expands to <code>\relax</code> if the entry hasn’t been defined.</td>
</tr>
</tbody>
</table>

---

§11.5.8; 572
\glsentrypdfsymbol{(entry-label)} glossaries-extra v1.42+

Used when \glossentrysymbol occurs in a PDF bookmark.

\Glsentryplural{(entry-label)} glossaries

Partially robust command that displays the value of the plural field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentryplural{(entry-label)} glossaries

Simply expands to the value of the plural field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the plural field doesn’t contain any fragile commands.

\glsentryprevcount{(entry-label)} glossaries v4.14+

Expands to the final entry count total from the previous \LaTeX run or if 0 if not available (needs to be enabled with \glsenableentrycount or \glsenableentryunitcount). With unit entry counting, this expands to the total for the current unit.

\glsentryprevmaxcount{(entry-label)} glossaries v4.14+

Expands to the maximum entry unit count total from the previous \LaTeX run or if 0 if not available (needs to be enabled with \glsenableentryunitcount).

\glsentryprevtotalcount{(entry-label)} glossaries v4.14+

Expands to the final entry count total from the previous \LaTeX run or if 0 if not available (needs to be enabled with \glsenableentryunitcount).
Command Summary

\Glsentryshort{⟨entry-label⟩}
glossaries v3.0+

Displays the value of the `short` field with sentence case applied. Does nothing if the entry hasn’t been defined. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentryshort{⟨entry-label⟩}
glossaries v3.0+

Simply expands to the value of the `short` field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the `short` field doesn’t contain any fragile commands.

\Glsentryshortpl{⟨entry-label⟩}
glossaries v3.0+

Displays the value of the `shortplural` field with sentence case applied. Does nothing if the entry hasn’t been defined. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentryshortpl{⟨entry-label⟩}
glossaries v3.0+

Simply expands to the value of the `shortplural` field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the `shortplural` field doesn’t contain any fragile commands.

\Glsentrysymbol{⟨entry-label⟩}
glossaries

Partially robust command that displays the value of the `symbol` field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentrysymbol{⟨entry-label⟩}
glossaries

709
Simply expands to the value of the `symbol` field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the `symbol` field doesn’t contain any fragile commands.

\glsentrysymbolaccess\{\entrylabel\} \glossaries-accsupp

As \glsentrysymbol but for the `symbolaccess` field.

\glsentrysymbolplural\{\entrylabel\} \glossaries

Simply expands to the value of the `symbolplural` field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the `symbolplural` field doesn’t contain any fragile commands.

\Glsentrytext\{\entrylabel\} \glossaries

Partially robust command that displays the value of the `text` field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentrytext\{\entrylabel\} \glossaries

Simply expands to the value of the `text` field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the `text` field doesn’t contain any fragile commands.

\glsentrytitlecase\{\entrylabel\}\{\fieldlabel\} \glossaries v4.22+

Applies title case to the value supplied in the given field (which is obtained with \glsfield-fetch).

\glsentrytype\{\entrylabel\} \glossaries
Command Summary

Simply expands to the value of the type key. Does nothing if the entry hasn’t been defined.

\Glsentryuseri{⟨entry-label⟩}  
\textit{glossaries v2.04+}

Partially robust command that displays the value of the user1 field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentryuseri{⟨entry-label⟩}  
\textit{glossaries v2.04+}

Simply expands to the value of the user1 field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the user1 field doesn’t contain any fragile commands.

\Glsentryuserii{⟨entry-label⟩}  
\textit{glossaries v2.04+}

Partially robust command that displays the value of the user2 field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentryuserii{⟨entry-label⟩}  
\textit{glossaries v2.04+}

Simply expands to the value of the user2 field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the user2 field doesn’t contain any fragile commands.

\Glsentryuseriii{⟨entry-label⟩}  
\textit{glossaries v2.04+}

Partially robust command that displays the value of the user3 field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\glsentryuseriii{⟨entry-label⟩}  
\textit{glossaries v2.04+}
### Command Summary

Simply expands to the value of the **user3** field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the **user3** field doesn’t contain any fragile commands.

\[
\texttt{\textbackslash Glsentryuseriv\{⟨entry-label⟩\}} \quad \text{glossaries v2.04+}
\]

Partially robust command that displays the value of the **user4** field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\[
\texttt{\textbackslash glsentryuseriv\{⟨entry-label⟩\}} \quad \text{glossaries v2.04+}
\]

Simply expands to the value of the **user4** field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the **user4** field doesn’t contain any fragile commands.

\[
\texttt{\textbackslash Glsentryuseriv\{⟨entry-label⟩\}} \quad \text{glossaries v2.04+}
\]

Partially robust command that displays the value of the **user5** field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\[
\texttt{\textbackslash glsentryuseriv\{⟨entry-label⟩\}} \quad \text{glossaries v2.04+}
\]

Simply expands to the value of the **user5** field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the **user5** field doesn’t contain any fragile commands.

\[
\texttt{\textbackslash Glsentryuseriv\{⟨entry-label⟩\}} \quad \text{glossaries v2.04+}
\]

Partially robust command that displays the value of the **user6** field with sentence case applied. As from glossaries v4.50, this command can expand in PDF bookmarks. Outside of PDF bookmarks it will expand to a robust internal command.

\[
\texttt{\textbackslash glsentryuseriv\{⟨entry-label⟩\}} \quad \text{glossaries v2.04+}
\]
Simply expands to the value of the \texttt{user6} field. Does nothing if the entry hasn’t been defined. May be used in expandable contexts provided that the \texttt{user6} field doesn’t contain any fragile commands.

Locally adds the field given by its internal field label \texttt{internal-field} to the inner formatting exclusion list for the entry identified by \texttt{entry-label}. This typically means that the field value already contains the inner formatting.

Expand field values when defining entries, except for those that explicitly have expansion disabled with \texttt{glssetnoexpandfield}.

A general purpose hook that’s performed within \texttt{glsxtrpostnamehook}.

If \texttt{glossaries-extra} has been loaded, this command will first check for the existence of the command \texttt{glsxtr\langle\texttt{category}\rangle\langle\texttt{field}\rangle\texttt{accsupp}}. If that command doesn’t exist or if \texttt{glossaries-extra} hasn’t been loaded, it then checks for the existence of \texttt{gls\langle\texttt{field}\rangle\texttt{accsupp}} (for example, \texttt{glsshortaccsupp}). Failing that it will use \texttt{glsaccsupp}. Whichever command is found first, \texttt{\langle\texttt{cs}\rangle\{\langle\texttt{replacement}\rangle\}}\{\langle\texttt{content}\rangle\} is performed.
Command Summary

Locally assigns the \langle value \rangle to the given field (identified by the internal field label \langle field \rangle) for the entry identified by \langle entry-label \rangle. Produces an error (or warning with \texttt{undefaction=warn}) if the entry or field doesn’t exist. Note that this doesn’t update any associated fields.

\glsfieldedef{\langle entry-label \rangle}{\langle field \rangle}{\langle value \rangle}

Locally assigns the full expansion of \langle value \rangle to the given field (identified by the internal field label \langle field \rangle) for the entry identified by \langle entry-label \rangle. Produces an error (or warning with \texttt{undefaction=warn}) if the entry or field doesn’t exist. Note that this doesn’t update any associated fields.

\glsfieldfetch{\langle entry-label \rangle}{\langle field-label \rangle}{\langle cs \rangle}

Fetches the value of the given field for the given entry and stores it in the command \langle cs \rangle. Triggers an error if the given field (identified by its internal field label) hasn’t been defined. Uses \glsdoifexists.

\glsfieldgdef{\langle entry-label \rangle}{\langle field \rangle}{\langle value \rangle}

As \glsfieldedef but does a global assignment.

\glsfieldxdef{\langle entry-label \rangle}{\langle field \rangle}{\langle value \rangle}

As \glsfieldedef but does a global assignment.

\glsFindWidestAnyName[\langle glossary labels \rangle]

Finds and sets the widest name for all entries in the given glossaries.

\glsFindWidestAnyNameLocation[\langle glossary labels \rangle]{\langle register \rangle]
Like `\glsFindWidestAnyName` but also also measures the location list. The length of the widest location is stored in ⟨register⟩, which should be a length register.

\begin{verbatim}
\glsFindWidestAnyNameSymbol[⟨glossary labels⟩]{⟨register⟩}
glossaries-extra-stylemods v1.05+
\end{verbatim}

§8.6.5.4; 436

Like `\glsFindWidestAnyNameSymbol` but also also measures the symbol. The length of the widest symbol is stored in ⟨register⟩ which should be a length register.

\begin{verbatim}
\glsFindWidestAnyNameSymbolLocation[⟨glossary labels⟩]{⟨register1⟩}{⟨register2⟩}
glossaries-extra-stylemods v1.05+
\end{verbatim}

§8.6.5.4; 436

Like `\glsFindWidestAnyNameSymbolLocation` but also also measures the location list. The length of the widest symbol is stored in ⟨register1⟩ and the length of the widest location is stored in ⟨register2⟩, which should both be length registers.

\begin{verbatim}
\glsFindWidestLevelTwo[⟨glossary labels⟩]
glossaries-extra-stylemods v1.05+
\end{verbatim}

§8.6.5.4; 436

Finds and sets the widest name for all entries with hierarchical level less than or equal to 2 in the given glossaries.

\begin{verbatim}
\glsFindWidestTopLevelName
glossaries-extra-stylemods
\end{verbatim}

§8.6.5.4; 435

A synonym for `\glsfindwidesttoplevelname`.

\begin{verbatim}
\glsfindwidesttoplevelname[⟨glossary labels⟩]
glossary-tree v4.22+
\end{verbatim}

§8.6.5.4; 435

Finds and sets the widest name for all top-level entries in the given glossaries. If the optional argument is omitted, the list of all non-ignored glossaries is assumed.

\begin{verbatim}
\glsFindWidestUsedAnyName[⟨glossary labels⟩]
glossaries-extra-stylemods v1.05+
\end{verbatim}

§8.6.5.4; 435
Command Summary

Finds and sets the widest name for all entries that have been marked as used in the given glossaries.

\texttt{\glsFindWidestUsedAnyNameLocation}[\langle\text{glossary labels}\rangle] \{\langle\text{register}\rangle\}
\texttt{glossaries-extra-stylemods v1.05+}$\S 8.6.5.4$; 436

Like $\glsFindWidestUsedAnyName$ but also also measures the location list. The length of the widest location is stored in $\langle\text{register}\rangle$, which should be a length register.

\texttt{\glsFindWidestUsedAnyNameSymbol}[\langle\text{glossary labels}\rangle] \{\langle\text{register}\rangle\}
\texttt{glossaries-extra-stylemods v1.05+}$\S 8.6.5.4$; 436

Like $\glsFindWidestUsedAnyName$ but also also measures the symbol. The length of the widest symbol is stored in $\langle\text{register}\rangle$ which should be a length register.

\texttt{\glsFindWidestUsedAnyNameSymbolLocation}[\langle\text{glossary labels}\rangle] \{\langle\text{register1}\rangle\} \{\langle\text{register2}\rangle\}
\texttt{glossaries-extra-stylemods v1.05+}$\S 8.6.5.4$; 436

Like $\glsFindWidestUsedAnyNameSymbol$ but also also measures the location list. The length of the widest symbol is stored in $\langle\text{register1}\rangle$ and the length of the widest location is stored in $\langle\text{register2}\rangle$, which should both be length registers.

\texttt{\glsFindWidestUsedLevelTwo}[\langle\text{glossary labels}\rangle]
\texttt{glossaries-extra-stylemods v1.05+}$\S 8.6.5.4$; 435

Finds and sets the widest name for all entries that have been marked as used with hierarchical level less than or equal to 2 in the given glossaries.

\texttt{\glsFindWidestUsedTopLevelName}[\langle\text{glossary labels}\rangle]
\texttt{glossaries-extra-stylemods v1.05+}$\S 8.6.5.4$; 435

Finds and sets the widest name for all top-level entries that have been marked as used in the given glossaries.
{
\textbf{Command Summary}

\GLSfirst[(\textit{options})]{(\textit{entry-label})}[{(\textit{insert})}]\text{ modifiers: * + 〈alt-mod〉} \text{ glossaries}

As \texttt{\textbackslash glsfirst} but converts the link text to all caps. If you have defined the entry with \texttt{\textbackslash newabbreviation} use \texttt{\GLSxtrfull} or \texttt{\GLS[prereset]} instead.

\Glsfirst[(\textit{options})]{(\textit{entry-label})}[{(\textit{insert})}]\text{ modifiers: * + 〈alt-mod〉} \text{ glossaries}

As \texttt{\textbackslash glsfirst} but converts the first character of the link text to uppercase (for the start of a sentence) using \texttt{\makefirstuc}. If you have defined the entry with \texttt{\textbackslash newabbreviation} use \texttt{\GLSxtrfull} or \texttt{\Glsp[prereset]} instead.

\glsfirst[(\textit{options})]{(\textit{entry-label})}[{(\textit{insert})}]\text{ modifiers: * + 〈alt-mod〉} \text{ glossaries}

References the entry identified by 〈\textit{entry-label}〉. The text produced is obtained from the \texttt{first} value. The 〈\textit{insert}〉 argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. If you have defined the entry with \texttt{\textbackslash newabbreviation} use \texttt{\GLSxtrfull} for the full form or \texttt{\GLSxtrlong} for the long form or use \texttt{\gls[prereset]}, as some abbreviation styles are too complicated to work with \texttt{\glsfirst}. For the first optional argument, see \texttt{\textbackslash glslink \textit{options}}.

\glsfirstabbrvdefaultfont{〈\textit{text}〉}

Formatting command for the short form on first use used by the abbreviation styles that don’t apply a font change by default.

\glsfirstabbrvemfont{〈\textit{text}〉} \text{ glossaries-extra v1.04+}

Short form font used by the “em” abbreviation styles on first use.

\glsfirstabbrvfont{〈\textit{text}〉}

Font formatting command for the short form on first use, initialised by the abbreviation style.
### Command Summary

- **\glsfirstabbrvhyphenfont{(text)}**  
  Short form font used by the “hyphen” abbreviation styles on first use.  
  - **glossaries-extra v1.17+**

- **\glsfirstabbrvonlyfont{(text)}**  
  Short form font used by the “only” abbreviation styles on first use.  
  - **glossaries-extra v1.17+**

- **\glsfirstabbrvscfont{(text)}**  
  Short form font used by the small caps “sc” abbreviation styles on first use.  
  - **glossaries-extra v1.17+**

- **\glsfirstabbrvsconlyfont{(text)}**  
  Short form font used by the “sc-only” abbreviation styles on first use.  
  - **glossaries-extra v1.48+**

- **\glsfirstabbrvscuserfont{(text)}**  
  Short form font used by the small caps “sc-user” abbreviation styles on first use.  
  - **glossaries-extra v1.48+**

- **\glsfirstabbrvsmfont{(text)}**  
  Short form font used by the “sm” abbreviation styles on first use.  
  - **glossaries-extra v1.17+**

- **\glsfirstabbrvuserfont{(text)}**  
  Short form font used by the “user” abbreviation styles on first use.  
  - **glossaries-extra v1.04+**

- **\glsfirstaccessdisplay{(text)}{(entry-label)}**  
  Short form font used by the “access-display” styles on first use.  
  - **glossaries-accsupp**

---

718
Does \textit{(text)} with the \texttt{firstaccess} replacement text (if set).

\begin{verbatim}
glsfirstinnerfmtabbrvfont{(text)} \glsfirstinnerfmtabbrvfont{(text)} \glsfirstinnerfmtlongfont{(text)} \glsfirstlongdefaultfont{(text)} \glsfirstlongemfont{(text)} \glsfirstlongfont{(text)} \glsfirstlongfootnotefont{(text)} \glsfirstlonghyphenfont{(text)}
\end{verbatim}

Applies both \glsfirstabbrvfont and \glsxtrgenentrytextfmt to \textit{(text)}.

\begin{verbatim}
glsfirstinnerfmtlongfont{(text)} \glsfirstinnerfmtlongfont{(text)}
\end{verbatim}

Applies both \glsfirstlongfont and \glsxtrgenentrytextfmt to \textit{(text)}.

Formatting command for the long form on first use used by the abbreviation styles that don’t apply a font change by default.

\begin{verbatim}
glsfirstlongemfont{(text)}
\end{verbatim}

Long form font used by the “em” abbreviation styles on first use.

\begin{verbatim}
glsfirstlongfont{(text)}
\end{verbatim}

Font formatting command for the long form on first use, initialised by the abbreviation style.

\begin{verbatim}
glsfirstlongfootnotefont{(text)}
\end{verbatim}

Formatting command for the first use long form used by the footnote abbreviation styles.

\begin{verbatim}
glsfirstlonghyphenfont{(text)}
\end{verbatim}

Long form font used by the “hyphen” abbreviation styles on first use.
## Command Summary

**\glsfirstlongonlyfont\{(text)\}**  
Glossaries-extra v1.17+

Long form font used by the “only” abbreviation styles on first use.

**\glsfirstlonguserfont\{(text)\}**  
Glossaries-extra v1.04+

Long form font used by the “user” abbreviation styles on first use.

**\GLSfirstplural\{(options)\}\{\langle entry-label\rangle\}\{\langle insert\rangle\}  modifiers: * + \langle alt-mod\rangle**  
Glossaries

As \glsfirstplural but converts the link text to all caps. If you have defined the entry with \newabbreviation use \GLSxtrfullpl or \GLSpl[prereset] instead.

**\Glsfirstplural\{(options)\}\{\langle entry-label\rangle\}\{\langle insert\rangle\}  modifiers: * + \langle alt-mod\rangle**  
Glossaries

As \glsfirstplural but converts the first character of the link text to uppercase (for the start of a sentence) using \makefirstuc. If you have defined the entry with \newabbreviation use \Glsxtrfullpl or \Glsp[prereset] instead.

**\glsfirstplural\{(options)\}\{\langle entry-label\rangle\}\{\langle insert\rangle\}  modifiers: * + \langle alt-mod\rangle**  
Glossaries

References the entry identified by \langle entry-label\rangle. The text produced is obtained from the firstplural value. The \langle insert\rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. If you have defined the entry with \newabbreviation use \glsxtrfullpl for the full form or \glsxtrlongpl for the long form or use \glsp[prereset], as some abbreviation styles are too complicated to work with \glsfirstplural. For the first optional argument, see \glslink options.

**\glsfirstpluralaccessdisplay\{(text)\}\{\langle entry-label\rangle\}**  
Glossaries-accsupp

Does \langle text\rangle with the firstpluralaccess replacement text (if set).
§4.5.3.1; 167

\glsfirstxpabbrvfont{\langle text\rangle}{\langle category\rangle} glossaries-extra v1.49+

If the markshortwords attribute is set for the given category, this encapsulates \langle text\rangle with \glsfirstabbrvfont otherwise with \glsfirstinnerfmtabbrvfont. This command has to expand, so protect any content that shouldn’t expand.

§4.5.3.1; 168

\glsfirstxplongfont{\langle category\rangle}{\langle text\rangle} glossaries-extra v1.49+

If the markwords attribute is set for the given category, this encapsulates \langle text\rangle with \glsfirstlongfont otherwise with \glsinnerfmtlongfont. This command has to expand, so protect any content that shouldn’t expand.

§5.5.3; 245

\GLSfmtfield{\langle insert\rangle}{\langle cs\rangle}{\langle entry-label\rangle}{\langle internal-field\rangle} glossaries-extra v1.49+

As \glsfmtfield but changes the field value to all caps.

§5.5.3; 245

\Glsfmtfield{\langle insert\rangle}{\langle cs\rangle}{\langle entry-label\rangle}{\langle internal-field\rangle} glossaries-extra v1.49+

As \glsfmtfield but uses \makefirstuc to change the field value to sentence case.

§5.5.3; 245

\glsfmtfield{\langle insert\rangle}{\langle cs\rangle}{\langle entry-label\rangle}{\langle internal-field\rangle} glossaries-extra v1.49+

Applies the formatting command \langle cs\rangle (which takes one argument) to the entry’s field value identified by the given internal field label, including \langle insert\rangle appended. Used by the inner formatting commands. Note that \glsfmtfield should not be robust as it needs to expand if it’s inside a case-changing command.

§5.3.2; 213

\GLSfmtfirst{\langle entry-label\rangle} glossaries-extra v1.42+

For use within captions or section titles to display the formatted all caps first.

§5.3.2; 213
Command Summary

For use within captions or section titles to display the formatted sentence case first.

\glsfmtfirst{⟨entry-label⟩}

§5.3.2; 213

For use within captions or section titles to display the formatted first.

\GLSfmtfirstpl{⟨entry-label⟩}
glossaries-extra v1.42+

§5.3.2; 214

For use within captions or section titles to display the formatted all caps firstplural.

\Glsfmtfirstpl{⟨entry-label⟩}

§5.3.2; 214

For use within captions or section titles to display the formatted sentence case firstplural.

\glsfmtfirstpl{⟨entry-label⟩}

§5.3.2; 213

For use within captions or section titles to display the formatted firstplural.

\GLSfmtfull{⟨entry-label⟩}
glossaries-extra v1.42+

§5.3.2; 211

Designed for use in section headings or captions, this expands to just \glspdfffmtfull{⟨entry-label⟩} in PDF bookmarks (no case-change), otherwise it expands to \Glsxtrtitlefull{⟨entry-label⟩}.

\GLSfmtfull{⟨entry-label⟩}
glossaries-extra v1.02+

§5.3.2; 211

Designed for use in section headings or captions, this expands to just \glspdfffmtfull{⟨entry-label⟩} in PDF bookmarks (no case-change), otherwise it expands to \Glsxtrtitlefull{⟨entry-label⟩}.

\glsfmtfull{⟨entry-label⟩}
glossaries-extra v1.02+

§5.3.2; 210
Command Summary

Designed for use in section headings or captions, this expands to just \glspdffmtfull{⟨entry-label⟩} in PDF bookmarks, otherwise it expands to \glsxtrtitlefull{⟨entry-label⟩}.

\GLSfmtfullpl{⟨entry-label⟩}  
glossaries-extra v1.42+

§5.3.2; 211

Designed for use in section headings or captions, this expands to just \glspdffmtfullpl{⟨entry-label⟩} in PDF bookmarks (no case-change), otherwise it expands to \GLSxtrtitlefullpl{⟨entry-label⟩}.

\Glsfmtfullpl{⟨entry-label⟩}  
glossaries-extra v1.02+

§5.3.2; 211

Designed for use in section headings or captions, this expands to just \glspdffmtfullpl{⟨entry-label⟩} in PDF bookmarks (no case-change), otherwise it expands to \Glsxtrtitlefullpl{⟨entry-label⟩}.

\glsfmtfullpl{⟨entry-label⟩}  
glossaries-extra v1.02+

§5.3.2; 211

Designed for use in section headings or captions, this expands to just \glspdffmtfullpl{⟨entry-label⟩} in PDF bookmarks, otherwise it expands to \glsxtrtitlefullpl{⟨entry-label⟩}.

\glsfmtfullpl{⟨entry-label⟩}  
glossaries-extra v1.02+

§5.3.2; 211

Designed for use in section headings or captions, this expands to just \glspdffmtfullpl{⟨entry-label⟩} in PDF bookmarks, otherwise it expands to \glsxtrtitlefullpl{⟨entry-label⟩}.

\GLSfmtinsert  
glossaries-extra v1.49+

As \glsfmtinsert but converts to all caps.

\glsfmtinsert  
glossaries-extra v1.49+

A shortcut that applies \glsxtrgenentrytextfmt to \glsinsert if \glsinsert isn’t empty.

\GLSfmtlong{⟨entry-label⟩}  

§5.3.2; 209

723
For use within captions or section titles to display the formatted all caps long form.

\texttt{\Glsfmtlong{⟨entry-label⟩}} \footnote{§5.3.2; 208}

For use within captions or section titles to display the formatted sentence case long form.

\texttt{\glsfmtlong{⟨entry-label⟩}} \footnote{§5.3.2; 208}

For use within captions or section titles to display the formatted long form.

\texttt{\GLSfmtlongpl{⟨entry-label⟩}} \footnote{§5.3.2; 209}

For use within captions or section titles to display the formatted all caps long plural form.

\texttt{\Glsfmtlongpl{⟨entry-label⟩}} \footnote{§5.3.2; 209}

For use within captions or section titles to display the formatted sentence case long plural form.

\texttt{\glsfmtlongpl{⟨entry-label⟩}} \footnote{§5.3.2; 209}

For use within captions or section titles to display the formatted long plural form.

\texttt{\GLSfmtname{⟨entry-label⟩}} \footnote{§5.3.2; 212}

For use within captions or section titles to display the formatted all caps name.

\texttt{\Glsfmtname{⟨entry-label⟩}} \footnote{§5.3.2; 212}

For use within captions or section titles to display the formatted sentence case name.
Command Summary

\glsfmtname{⟨entry-label⟩}  
For use within captions or section titles to display the formatted name.

\GLSfmtplural{⟨entry-label⟩}  
For use within captions or section titles to display the formatted all caps plural.

\Glsfmtplural{⟨entry-label⟩}  
For use within captions or section titles to display the formatted sentence case plural.

\glsfmtplural{⟨entry-label⟩}  
For use within captions or section titles to display the formatted plural.

\GLSfmtshort{⟨entry-label⟩}  
For use within captions or section titles to display the formatted all caps short form.

\Glsfmtshort{⟨entry-label⟩}  
For use within captions or section titles to display the formatted sentence case short form.

\glsfmtshort{⟨entry-label⟩}  
For use within captions or section titles to display the formatted short form.

\GLSfmtshortpl{⟨entry-label⟩}  
For use within captions or section titles to display the formatted short form.
Command Summary

For use within captions or section titles to display the formatted all caps short plural form.

\texttt{\textbackslash Glstfmtshortpl\{\textlangle entry-label\}\}}

For use within captions or section titles to display the formatted sentence case short plural form.

\texttt{\textbackslash glstfmtshortpl\{\textlangle entry-label\}\}}

For use within captions or section titles to display the formatted short plural form.

\texttt{\textbackslash GLSfmttext\{\textlangle entry-label\}\}}

For use within captions or section titles to display the formatted all caps text.

\texttt{\textbackslash Glstfmttext\{\textlangle entry-label\}\}}

For use within captions or section titles to display the formatted sentence case text.

\texttt{\textbackslash glstfmttext\{\textlangle entry-label\}\}}

For use within captions or section titles to display the formatted text.

\texttt{\textbackslash glstforeachincategory\{\langle glossary-types\}\{\langle category\}\{\langle glossary-cs\}\{\langle label-cs\}\{\langle body\}\}}

Iterates over all entry in the given list of glossaries (or all non-ignored glossaries, if the optional argument is omitted) and performs \textit{body} for those entries that have the category set to \textit{category}. Within \textit{body}, the current entry can be referenced with \textit{label-cs} and the glossary can be referenced with \textit{glossary-cs}.
Command Summary

\texttt{\glsgenentryfmt} \quad \text{glossaries v3.11a+}

The display format used by \texttt{\glssentryfmt} for regular entries.

\texttt{\glsgetattribute}\{\texttt{(entry-label)}\}\{\texttt{(attribute)}\}

Expands to the value of the given attribute for the category associated with the entry identified by \texttt{(entry-label)}. Expands to nothing if the attribute hasn’t been set.

\texttt{\glsgetcategoryattribute}\{\texttt{(category)}\}\{\texttt{(attribute)}\}

Expands to the value of the given attribute for the given category. Expands to nothing if the attribute hasn’t been set.

\texttt{\glsgetwidestname} \quad \text{glossaries-extra-stylemods v1.05+}

Expands to the widest top-level name.

\texttt{\glsgetwidestsubname}\{\texttt{(level)}\} \quad \text{glossaries-extra-stylemods v1.05+}

Expands to the widest name for the given hierarchical level or to the widest top-level name, if no widest name set for \texttt{(level)}.

\texttt{\glsgroupheading}\{\texttt{(group-label)}\} \quad \text{glossaries}

Iterates over all entry in the given list of glossaries (or all non-ignored glossaries, if the optional argument is omitted) and performs \texttt{(body)} for those entries that have the attribute given by \texttt{(attribute-label)} set to \texttt{(attribute-value)}. Within \texttt{(body)}, the current entry can be referenced with \texttt{(label-cs)} and the glossary can be referenced with \texttt{(glossary-cs)}.
Command Summary

Inserted at the start of each group in a glossary (unless \texttt{groups=false}) to display the group’s heading, if applicable, using the title associated with ⟨\texttt{group-label}⟩ or, if no title provided, just ⟨\texttt{group-label}⟩. This command is defined by glossary styles as appropriate.

\texttt{\texttt{\textbackslash glsgroupskip}}

Inserted before each group heading (except the first) in a glossary (unless \texttt{groups=false}). This command is defined by glossary styles as appropriate. Most of the predefined styles define this command to check the \texttt{nogroupskip} option.

\texttt{\textbackslash glshasattribute}{⟨\texttt{entry-label}⟩}{⟨\texttt{attribute}⟩}{⟨\texttt{true}⟩}{⟨\texttt{false}⟩}

Tests if the given attribute has been set for the category associated with the entry identified by ⟨\texttt{entry-label}⟩ (using etoolbox’s \texttt{\textbackslash ifcsvoid}). Does ⟨\texttt{false}⟩ if the entry hasn’t been defined.

\texttt{\textbackslash glshascategoryattribute}{⟨\texttt{category}⟩}{⟨\texttt{attribute}⟩}{⟨\texttt{true}⟩}{⟨\texttt{false}⟩}

Tests if the given attribute has been set for the given category (using etoolbox’s \texttt{\textbackslash ifcsvoid}).

\texttt{\textbackslash glashashchar}

Expands to a literal hash #.

\texttt{\textbackslash glshex}{⟨\texttt{hex}⟩}

Expands to \texttt{\textbackslash string\textbackslash u(\texttt{hex})}.

\texttt{\textbackslash glshyperlink}{⟨\texttt{text}⟩}{⟨\texttt{entry-label}⟩}

Creates a hyperlink to the given entry with the hyperlink text provided in the optional argument. If omitted, the default is \texttt{\textbackslash glsentrytext}{⟨\texttt{entry-label}⟩}.
Command Summary

\glshypernumber\{\langle location(s)\rangle\}
glossaries

This will encapsulate each location with a hyperlink, if supported. This may be used as a location encap. The argument may be a single location or locations delimited by \delimR or \delimN. This command should not be used outside of location lists as it requires additional information in order to correctly form the hyperlinks.

\glsifapplyinnerfmtfield\{\langle entry-label\rangle\}\{\langle internal-field\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra v1.49+

Tests if the field given by its internal field label \langle internal-field\rangle has been added to the inner formatting exclusion list for the entry identified by \langle entry-label\rangle.

\glsifattribute\{\langle entry-label\rangle\}\{\langle attribute\rangle\}\{\langle value\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
§10.2.2; 505

glossaries-extra v1.49+

Tests if the category associated with the entry identified by \langle entry-label\rangle has the given attribute set to \langle value\rangle. Does \langle true\rangle if the attribute is \langle value\rangle and \langle false\rangle otherwise. Does \langle false\rangle if there’s no such attribute for the given category or if the entry hasn’t been defined.

\glsifattributetrue\{\langle entry-label\rangle\}\{\langle attribute\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
§10.2.2; 506

glossaries-extra v1.49+

Tests if the category associated with the entry given by \langle entry-label\rangle has the given attribute set to true. Does \langle true\rangle if the attribute is true and \langle false\rangle otherwise. Does \langle false\rangle if there’s no such attribute for the given category or if the entry hasn’t been defined.

\glsifcategory\{\langle entry-label\rangle\}\{\langle category\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
§10; 492

Tests if the entry identified by \langle entry-label\rangle has the category set to \langle category\rangle (uses \if\glsgsfieldeq for the test).

\glsifcategoryattribute\{\langle category\rangle\}\{\langle attribute\rangle\}\{\langle value\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
§10.2.2; 505
Tests if the given category has the given attribute set to \langle value \rangle. Does \langle true \rangle if the attribute is \langle value \rangle and \langle false \rangle otherwise. Does \langle false \rangle if there’s no such attribute for the given category.

\glsifcategoryattributehasitem\{\langle category \rangle\}\{\langle attribute \rangle\}\{\langle item \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}
§10.2.2; 507

glossaries-extra v1.49+

Does \langle true \rangle if the category has the attribute (whose value is a comma-separated list) contains the given item and \langle false \rangle otherwise. Does \langle false \rangle if there’s no such attribute for the given category. The item and list are expanded and passed to datatool’s \DTLifinlist to perform the test.

\glsifcategoryattributetrue\{\langle category \rangle\}\{\langle attribute \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}
§10.2.2; 506

glossaries-extra v1.49+

Tests if the given category has the given attribute set to \true. Does \true if the attribute is \true and \false otherwise. Does \false if there’s no such attribute for the given category.

\glsifindexed\{\langle entry-label \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}
§5.8; 264

glossaries-extra v1.49+

Tests if the value obtained from \glsentryindexcount is greater than 0.

\glsifnotregular\{\langle entry-label \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}
§10.2.2; 506

glossaries-extra v1.04+

Does \true if the category for the given entry has the \regular attribute explicitly set to \false, otherwise does \false.

\glsifnotregularcategory\{\langle category \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}
§10.2.2; 506

glossaries-extra v1.04+

Does \true if the given category has the \regular attribute explicitly set to \false, otherwise does \false.

\glsifplural\{\langle true \rangle\}\{\langle false \rangle\}
glossaries
Initialised by the \gls{like} and \glstext{like} commands, this expands to \textit{true} if the calling command accesses a plural field (such as \glspl{} or \glsp{}ural) otherwise it expands to \textit{false}. This command may be used within associated hooks, entry display styles (\defgls{entryfmt}), and the post-link hook.

\begin{Verbatim}
\glsifregular\{\textit{entry-label}\}\{\textit{true}\}\{\textit{false}\}
\end{Verbatim}

Does \textit{true} if the category for the given entry has the \texttt{regular} attribute explicitly set to \textit{true}, otherwise does \textit{false}.

\begin{Verbatim}
\glsifregularcategory\{\textit{category}\}\{\textit{true}\}\{\textit{false}\}
\end{Verbatim}

Does \textit{true} if the given category has the \texttt{regular} attribute explicitly set to \textit{true}, otherwise does \textit{false}.

\begin{Verbatim}
\glsignore\{\textit{text}\}
\end{Verbatim}

Does nothing. When used as a location encap, this signifies to \texttt{bib2gls} that the entry is required but the location shouldn’t be added to the location list. With other indexing methods, this simply creates an invisible location.

\begin{Verbatim}
\glsindexingsetting
\end{Verbatim}

Indicates what indexing option has been chosen.

\begin{Verbatim}
\glsindexsubgroupitem\{\textit{previous group level}\}\{\textit{level}\}\{\textit{parent label}\}\{\textit{group label}\}\{\textit{group title}\}
\end{Verbatim}

Used to format sub-group headers for the \texttt{indexgroup} styles.

\begin{Verbatim}
\glsinitreunsets
\end{Verbatim}

Indicates what indexing option has been chosen.
Hook that initialises the \texttt{prerese}, \texttt{preunset} and \texttt{postunset} settings.

\begin{verbatim}
\texttt{\glsinlinedescformat\langle description\rangle\langle symbol\rangle\langle location list\rangle}
glossary-inline v3.03+
\end{verbatim}

Formats the description, symbol and location list for top-level entries.

\begin{verbatim}
\texttt{\glsinlinesubdescformat\langle description\rangle\langle symbol\rangle\langle location list\rangle}
glossary-inline v3.03+
\end{verbatim}

Formats the description, symbol and location list for child entries.

\begin{verbatim}
\texttt{\glsinnerfmtabbrvfont\langle text\rangle}
glossaries-extra v1.49+
\end{verbatim}

Robust command that applies both \texttt{\glsabbrvfont} and \texttt{\glsxtrgenentrytextfmt} to \texttt{\langle text\rangle}.

\begin{verbatim}
\texttt{\glsinnerfmtlongfont\langle text\rangle}
glossaries-extra v1.49+
\end{verbatim}

Applies both \texttt{\glslongfont} and \texttt{\glsxtrgenentrytextfmt} to \texttt{\langle text\rangle}.

\begin{verbatim}
\texttt{\glsinsert}
glossaries
\end{verbatim}

The final \texttt{\langle insert\rangle} argument passed to the \texttt{\gls-like} commands (but not to the \texttt{\glslike} commands, where the \texttt{\langle insert\rangle} is added to \texttt{\glslikecustomtext}). This command may be used within associated hooks, entry display styles (\texttt{\defglsentryfmt}), and the post-link hook.

\begin{verbatim}
\texttt{\glskeylisttok}
glossaries
\end{verbatim}

A token register that stores the options passed to \texttt{\newabbreviation}.

\begin{verbatim}
\texttt{\glslabel}
glossaries
\end{verbatim}

732
**Command Summary**

The current entry label, initialised by the \gls-like and \glstext-like commands. This command may be used within associated hooks, entry display styles (\defglsentryfmt), and the post-link hook.

\glslabeltok glossaries §4.5.3.1; 163

A token register that stores the entry’s label.

\glsetentryfield{(cs)\{⟨entry-label⟩\}⟨field-label⟩} glossaries v4.07+

Fetches the value of the given field (identified by its internal label ⟨field-label⟩) for the entry given by ⟨entry-label⟩ and stores it in the command ⟨cs⟩.

\Glslink[⟨options⟩]{⟨entry-label⟩}{⟨text⟩} modifiers: * + ⟨alt-mod⟩ glossaries v4.50+

As \glslink but sets the link text to \glssentencecase{⟨text⟩}. This is provided to allow a sentence case mapping in the event that \glslink occurs at the start of content that has automated case-changing.

\glslink[⟨options⟩]{⟨entry-label⟩}{⟨text⟩} modifiers: * + ⟨alt-mod⟩ glossaries

References the entry identified by ⟨entry-label⟩ with the given ⟨text⟩ as the link text. This command does not alter or depend on the first use flag (use \glsdisp instead, if the first use flag needs to be unset). This command is considered a \glstext-like command. For the first optional argument, see \glslink options.

\glslinkcheckfirsthyperhook glossaries v4.08+

Hook used at the end of the code in the \gls-like commands that tests if the hyperlink should be switched off on first use.

\glslinkpostsetkeys glossaries v4.16+

733
Hook implemented after setting the options passed to the \gls-like and \glstext-like commands.

\texttt{\textbackslash glslinkpresetkeys} \hspace{1cm} \textit{glossaries-extra v1.26+} \hspace{1cm} \S5.1.1; 188

Hook implemented before setting the options passed to the \gls-like and \glstext-like commands.

\texttt{\textbackslash glslinkwrccontent\{\langle code\rangle\}} \hspace{1cm} \textit{glossaries-extra v1.48+} \hspace{1cm} \S5; 185

Encapsulates the link text and indexing. Just does \langle code\rangle by default.

\texttt{\textbackslash glslistchildpostlocation \hspace{1cm} \textit{initial: .} \hspace{1cm} \textit{glossaries-extra-stylemods v1.21+}} \hspace{1cm} \S8.6.5.3; 429

Used after the child entry location list for the list styles.

\texttt{\textbackslash glslistchildprelocation \hspace{1cm} \textit{initial: \textbackslash glslistprelocation}} \hspace{1cm} \textit{glossaries-extra-stylemods v1.21+} \hspace{1cm} \S8.6.5.3; 428

Used before the child entry location list for the list styles.

\texttt{\textbackslash glslistdesc\{\langle entry-label\rangle\}} \hspace{1cm} \textit{glossaries-extra-stylemods v1.31+} \hspace{1cm} \S8.6.5.3; 429

Used to display the description for the list styles.

\texttt{\textbackslash glslistdottedwidth} \hspace{1cm} \textit{glossary-list}

A length register used by listdotted.

\texttt{\textbackslash glslistexpandedname\{\langle entry-label\rangle\}} \hspace{1cm} \textit{glossary-list v4.48+}
Command Summary

Used by \glslistinit to provide better integration with \gettitlestring.

\glslistgroupafterheader glossaries-extra-stylemods v1.47+ §8.6.5.3; 429

Used after group headings in the listgroup styles.

\glslistgroupheaderitem{⟨group-label⟩}{⟨header code⟩} glossaries-extra-stylemods v1.47+ §8.6.5.3; 429

Used to display the group headings in the listgroup styles.

\glslistgroupskip glossaries-extra-stylemods v1.31+ §8.6.5.3; 430

Used for the group skip in the list styles.

\glslistinit glossary-list v4.48+ §8.6.5.3; 429

Used to disable problematic commands at the start the list styles to provide better integration with \gettitlestring.

\glslistitem{⟨entry-label⟩} glossaries-extra-stylemods v1.47+ §8.6.5.3; 429

Used to display the top-level entry item in the list styles.

\glslistprelocation initial: \glsxtrprelocation glossaries-extra-stylemods v1.21+ §8.6.5.3; 428

Used before the top-level entry location list for the list styles.

\glslocalreset{⟨entry-label⟩} glossaries 735
Command Summary

Locally resets the entry’s first use flag. That is, this marks the entry as “not used”.

\texttt{\textbackslash glslocalresetall}\langle\textit{glossary labels list}\rangle \hspace{1cm} \textit{glossaries}

Locally resets the first use flag for all entries in whose labels are listed in the \langle\textit{glossary labels list}\rangle comma-separated list. If the optional argument is omitted, the list of all non-ignored glossaries is assumed.

\texttt{\textbackslash glslocalreseteach}\langle\textit{entry-labels}\rangle \hspace{1cm} \textit{glossaries-extra v1.31+}

Locally resets each listed entry’s first use flag.

\texttt{\textbackslash glslocalunset}\langle\textit{entry-label}\rangle \hspace{1cm} \textit{glossaries}

Locally unsets the entry’s first use flag. That is, this marks the entry as “used”.

\texttt{\textbackslash glslocalunsetall}\langle\textit{glossary labels list}\rangle \hspace{1cm} \textit{glossaries}

Locally unsets the first use flag for all entries in whose labels are listed in the \langle\textit{glossary labels list}\rangle comma-separated list. If the optional argument is omitted, the list of all non-ignored glossaries is assumed.

\texttt{\textbackslash glslocalunseteach}\langle\textit{entry-labels}\rangle \hspace{1cm} \textit{glossaries-extra v1.31+}

Locally unsets each listed entry’s first use flag.

\texttt{\textbackslash glslongaccessdisplay}\langle\textit{text}\rangle\langle\textit{entry-label}\rangle \hspace{1cm} \textit{glossaries-accsupp}

Does \textit{text} with the \texttt{longaccess} replacement text (if set).

\texttt{\textbackslash glslongdefaultfont}\langle\textit{text}\rangle \hspace{1cm} \textit{glossaries-extra v1.04+}
Formatting command for the long form used by the abbreviation styles that don’t apply a font change by default.

\glslongemfont\{⟨text⟩\}
glossaries-extra v1.04+

Long form font used by the “em” abbreviation styles.

\glslongextraDescAlign
glossary-longextra v1.37+

§8.7.2; 446

The horizontal alignment for the description column.

\glslongextraDescFmt\{⟨entry-label⟩\}
glossary-longextra v1.37+

§8.7.2; 446

Used by the glossary-longextra styles to display a top-level entry’s description and post-description hook.

\glslongextraDescNameHeader
glossary-longextra v1.37+

§8.7.2.1; 450

Sets the header and footer for the long-desc-name style with longtable.

\glslongextraDescNameTabularFooter
glossary-longextra v1.37+

§8.7.2.1; 450

Displays the footer for the long-desc-name style.

\glslongextraDescNameTabularHeader
glossary-longextra v1.37+

§8.7.2.1; 450

Displays the header for the long-desc-name style.

\glslongextraDescSymHeader
glossary-longextra v1.49+

§8.7.2.5; 458

Sets the header and footer for the long-desc-sym style with longtable.
Sets the header and footer for the long-desc-sym-name style with longtable.

Displays the footer for the long-desc-sym-name style.

Displays the header for the long-desc-sym-name style.

Displays the footer for the long-desc-sym style.

Displays the header for the long-desc-sym style.

Formats the top-level group heading.

Used to format the column headers.
Command Summary

The horizontal alignment for the location list column.

\texttt{\textbackslash glslongextraLocationDescNameHeader} \hspace{1cm} \texttt{glossary-longextra v1.37+} §8.7.2.3; 453

Sets the header and footer for the \texttt{long-loc-desc-name} style with \texttt{longtable}.

\texttt{\textbackslash glslongextraLocationDescNameTabularFooter} \hspace{1cm} \texttt{glossary-longextra v1.37+} §8.7.2.3; 453

Displays the footer for the \texttt{long-loc-desc-name} style.

\texttt{\textbackslash glslongextraLocationDescNameTabularHeader} \hspace{1cm} \texttt{glossary-longextra v1.37+} §8.7.2.3; 453

Displays the header for the \texttt{long-loc-desc-name} style.

\texttt{\textbackslash glslongextraLocationDescSymNameHeader} \hspace{1cm} \texttt{glossary-longextra v1.37+} §8.7.2.4; 456

Sets the header and footer for the \texttt{long-loc-desc-sym-name} style with \texttt{longtable}.

\texttt{\textbackslash glslongextraLocationDescSymNameTabularFooter} \hspace{1cm} \texttt{glossary-longextra v1.37+} §8.7.2.4; 456

Displays the footer for the \texttt{long-loc-desc-sym-name} style.

\texttt{\textbackslash glslongextraLocationDescSymNameTabularHeader} \hspace{1cm} \texttt{glossary-longextra v1.37+} §8.7.2.4; 455

Displays the header for the \texttt{long-loc-desc-sym-name} style.

\texttt{\textbackslash glslongextraLocationFmt\{\langle entry-label\rangle\}\{\langle location list\rangle\}} \hspace{1cm} \texttt{glossary-longextra v1.37+} §8.7.2; 447

Used by the \texttt{glossary-longextra} styles to display a top-level entry's location list.

739
Sets the header and footer for the long-loc-sym-desc-name style with \texttt{longtable}.

Displays the footer for the long-loc-sym-desc-name style.

Displays the header for the long-loc-sym-desc-name style.

Computes the value of \texttt{\glsdescwidth} according to the widest name for styles that only show the name, location list and description.

The formatting for the long form in the abbr-long-short and abbr-short-long styles.

The long column header for the abbr-long-short and abbr-short-long styles.

Sets the header and footer for the abbr-short-long style with \texttt{longtable}.
**Command Summary**

Displays the footer for the abbr–short–long style.

\glslongextraLongShortTabularHeader  
\textit{glossary–longextra v1.49+}  
§8.7.2.6; 461

Displays the header for the abbr–short–long style.

\glslongextraNameAlign  
\textit{initial: 1}  
\textit{glossary–longextra v1.37+}  
§8.7.2; 446

The horizontal alignment for the name column.

\glslongextraNameDescHeader  
\textit{glossary–longextra v1.37+}  
§8.7.2.1; 449

Sets the header and footer for the long–name–desc style with \texttt{longtable}.

\glslongextraNameDescLocationHeader  
\textit{glossary–longextra v1.37+}  
§8.7.2.3; 453

Sets the header and footer for the long–name–desc–loc style with \texttt{longtable}.

\glslongextraNameDescLocationTabularFooter  
\textit{glossary–longextra v1.37+}  
§8.7.2.3; 453

Displays the footer for the long–name–desc–loc style.

\glslongextraNameDescLocationTabularHeader  
\textit{glossary–longextra v1.37+}  
§8.7.2.3; 453

Displays the header for the long–name–desc–loc style.

\glslongextraNameDescSymHeader  
\textit{glossary–longextra v1.37+}  
§8.7.2.2; 451

Sets the header and footer for the long–name–desc–sym style with \texttt{longtable}.

741
Sets the header and footer for the long-name-desc-sym-loc style with longtable.

Displays the footer for the long-name-desc-sym-loc style.

Displays the header for the long-name-desc-sym-loc style.

Displays the footer for the long-name-desc-sym style.

Displays the header for the long-name-desc-sym style.

Displays the footer for the long-name-desc style.

Displays the header for the long-name-desc style.

\glslongextraNameFmt\{\entrylabel\}
Used by the glossary-longextra styles to add the hypertarget (if supported) and display a top-level entry’s name.

\texttt{\glslongextraNameSymDescHeader} \hspace{1em} \texttt{glossary-longextra v1.37+} \hspace{1em} §§8.7.2.2; 451

Sets the header and footer for the long-name-sym-desc style with longtable.

\texttt{\glslongextraNameSymDescLocationHeader} \hspace{1em} \texttt{glossary-longextra v1.37+} \hspace{1em} §§8.7.2.4; 455

Sets the header and footer for the long-name-sym-desc-loc style with longtable.

\texttt{\glslongextraNameSymDescLocationTabularFooter} \hspace{1em} \texttt{glossary-longextra v1.37+} \hspace{1em} §§8.7.2.4; 455

Displays the footer for the long-name-sym-desc-loc style.

\texttt{\glslongextraNameSymDescLocationTabularHeader} \hspace{1em} \texttt{glossary-longextra v1.37+} \hspace{1em} §§8.7.2.4; 454

Displays the header for the long-name-sym-desc-loc style.

\texttt{\glslongextraNameSymDescTabularFooter} \hspace{1em} \texttt{glossary-longextra v1.37+} \hspace{1em} §§8.7.2.2; 451

Displays the footer for the long-name-sym-desc style.

\texttt{\glslongextraNameSymDescTabularHeader} \hspace{1em} \texttt{glossary-longextra v1.37+} \hspace{1em} §§8.7.2.2; 451

Displays the header for the long-name-sym-desc style.

\texttt{\glslongextraSetDescWidth} \hspace{1em} \texttt{glossary-longextra v1.37+} \hspace{1em} §§8.7.2.1; 449

Computes the value of \texttt{\glsdescwidth} according to the widest name for styles that only show the name and description.
## Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glslongextraSetWidest{\langle\text{widest-name}\rangle}</code></td>
<td>Identifies \langle\text{widest-name}\rangle as the widest top-level name.</td>
<td>447</td>
</tr>
<tr>
<td><code>\glslongextraShortHeader</code></td>
<td>The short column header for the abbr-long-short and abbr-short-long styles.</td>
<td>458</td>
</tr>
<tr>
<td><code>\glslongextraShortLongHeader</code></td>
<td>Sets the header and footer for the abbr-short-long style with longtable.</td>
<td>460</td>
</tr>
<tr>
<td><code>\glslongextraShortLongTabularFooter</code></td>
<td>Displays the footer for the abbr-short-long style.</td>
<td>460</td>
</tr>
<tr>
<td><code>\glslongextraShortLongTabularHeader</code></td>
<td>Displays the header for the abbr-short-long style.</td>
<td>460</td>
</tr>
<tr>
<td><code>\glslongextraShortNoNameSetDescWidth</code></td>
<td>Sets the value of \texttt{\glssdescwidth} for the abbr-long-short and abbr-short-long styles.</td>
<td>460</td>
</tr>
<tr>
<td><code>\glslongextraShortTargetFmt\{\langle\text{entry-label}\rangle\}</code></td>
<td>The formatting, including the target, for the short form in the abbr-long-short and abbr-short-long styles.</td>
<td>459</td>
</tr>
<tr>
<td><code>\glslongextraSubDescFmt\{\langle\text{level}\rangle\}\{\langle\text{entry-label}\rangle\}</code></td>
<td></td>
<td>446</td>
</tr>
</tbody>
</table>
**Command Summary**

Used by the glossary-longextra styles to display a child entry’s description and post-description hook.

\[ \text{\texttt{\textbackslash glslongextraSubGroupHeading\{\langle number columns\}\{\langle prev group level\}\{\langle group level\}\{\langle parent-entry-label\}\{\langle group-label\}\}}] \]  
  glossary-longextra v1.49+

Forms the sub-group heading, if supported.

\[ \text{\texttt{\textbackslash glslongextraSubLocationFmt\{\langle level\}\{\langle entry-label\}\{\langle location list\}\}}} \]  
  glossary-longextra v1.37+

Used by the glossary-longextra styles to display a child entry’s location list.

\[ \text{\texttt{\textbackslash glslongextraSubLongFmt\{\langle level\}\{\langle entry-label\}\}}} \]  
  glossary-longextra v1.49+

The formatting for child entry long forms in the abbr-long-short and abbr-short-long styles.

\[ \text{\texttt{\textbackslash glslongextraSubNameFmt\{\langle level\}\{\langle entry-label\}\}}} \]  
  glossary-longextra v1.37+

Used by the glossary-longextra styles to add the hypertarget (if supported) for child-entries. The name isn’t shown by default.

\[ \text{\texttt{\textbackslash glslongextraSubShortTargetFmt\{\langle level\}\{\langle entry-label\}\}}} \]  
  glossary-longextra v1.49+

The formatting, including the target, for child entry short forms in the abbr-long-short and abbr-short-long styles.

\[ \text{\texttt{\textbackslash glslongextraSubSymbolFmt\{\langle level\}\{\langle entry-label\}\}}} \]  
  glossary-longextra v1.37+

Used by the glossary-longextra styles to display a child entry’s symbol.
### Command Summary

\texttt{\textbackslash glslongextraSubSymbolOrName\{\langle level\rangle\}\{\langle entry-label\rangle\}} \quad \textit{glossary-longextra v1.49+}

Adds the hypertarget (if supported) and displays the symbol if set or the name otherwise for child entries.

\texttt{\textbackslash glslongextraSubSymbolTargetFmt\{\langle level\rangle\}\{\langle entry-label\rangle\}}

\textit{glossary-longextra v1.49+}

Adds the hypertarget (if supported) and displays the symbol for child entries.

\texttt{\textbackslash glslongextraSymbolAlign} \quad \textit{initial: c} \quad \textit{glossary-longextra v1.37+}

The horizontal alignment for the symbol column.

\texttt{\textbackslash glslongextraSymbolFmt\{\langle entry-label\rangle\}}

\textit{glossary-longextra v1.37+}

Used by the \texttt{glossary-longextra} styles to display a top-level entry’s symbol.

\texttt{\textbackslash glslongextraSymbolNameAlign} \quad \textit{initial: l} \quad \textit{glossary-longextra v1.49+}

The horizontal alignment for the symbol column when it’s being used instead of the name.

\texttt{\textbackslash glslongextraSymbolOrName\{\langle entry-label\rangle\}}

\textit{glossary-longextra v1.49+}

Adds the hypertarget (if supported) and displays the symbol if set or the name otherwise for top-level entries.

\texttt{\textbackslash glslongextraSymbolTargetFmt\{\langle entry-label\rangle\}}

\textit{glossary-longextra v1.49+}

Adds the hypertarget (if supported) and displays the symbol for top-level entries.

746
Sets the header and footer for the long-sym-desc style with longtable.

Sets the header and footer for the long-sym-desc-name style with longtable.

Displays the footer for the long-sym-desc-name style.

Displays the header for the long-sym-desc-name style.

Displays the footer for the long-sym-desc style.

Displays the header for the long-sym-desc style.

Computes the value of $\text{glsdescwidth}$ according to the widest name for styles that show the name, symbol, location list and description.
Command Summary

Computes the value of \texttt{\glslongextraSymSetDescWidth} according to the widest name for styles that only show the symbol and description.

\texttt{\glslongextraSymSetDescWidth} \quad \textit{glossary-longextra v1.37+} \quad \S\S\ 8.7.2; 450

Computes the value of \texttt{\glslongextraSymSetDescWidth} according to the widest name for styles that only show the name, symbol and description.

\texttt{\glslongextraTabularVAlign} \quad \textit{initial: c} \quad \textit{glossary-longextra v1.37+} \quad \S\S\ 8.7.2; 445

Only for use with the \texttt{tabular} setting, this should expand to the \texttt{tabular} environment’s vertical alignment specifier.

\texttt{\glslongextraUpdateWidest\{\textit{name}\}} \quad \textit{glossary-longextra v1.37+} \quad \S\S\ 8.7.2; 447

If \textit{name} is wider than the current widest name, it will be set as the new widest name.

\texttt{\glslongextraUpdateWidestChild\{\textit{level}\}\{\textit{name}\}} \quad \textit{glossary-longextra v1.37+} \quad \S\S\ 8.7.2; 447

As \texttt{\glslongextraUpdateWidest} but for child entries. Does nothing by default.

\texttt{\GlsLongExtraUseTabularfalse} \quad \textit{glossary-longextra v1.37+} \quad \S\S\ 8.7.2; 445

Sets \texttt{\ifGlsLongExtraUseTabular} to false (if this setting is required, the style must be set after this command).

\texttt{\GlsLongExtraUseTabulartrue} \quad \textit{glossary-longextra v1.37+} \quad \S\S\ 8.7.2; 445

Sets \texttt{\ifGlsLongExtraUseTabular} to true (if this setting is required, the style must be set after this command).
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glslongfont{(text)}</td>
<td>Font formatting command for the long form, initialised by the abbreviation style.</td>
<td></td>
</tr>
<tr>
<td>\glslongfootnotefont{(text)}</td>
<td>Formatting command for the long form used by the footnote abbreviation styles.</td>
<td>glossaries-extra v1.05+</td>
</tr>
<tr>
<td>\glslonghyphenfont{(text)}</td>
<td>Long form font used by the “hyphen” abbreviation styles.</td>
<td>glossaries-extra v1.17+</td>
</tr>
<tr>
<td>\glslongonlyfont{(text)}</td>
<td>Long form font used by the “only” abbreviation styles.</td>
<td>glossaries-extra v1.17+</td>
</tr>
<tr>
<td>\glslongpltok</td>
<td>A token register that stores the long plural form (which may have been modified after being passed to \newabbreviation).</td>
<td>§4.5.3.1; 164</td>
</tr>
<tr>
<td>\glslongpluralaccessdisplay{(text)}{(entry-label)}</td>
<td>Does (text) with the longpluralaccess replacement text (if set).</td>
<td>glossaries-accsupp</td>
</tr>
<tr>
<td>\glslongtok</td>
<td>A token register that stores the long form (which may have been modified after being passed to \newabbreviation).</td>
<td>§4.5.3.1; 164</td>
</tr>
<tr>
<td>\glslonguserfont{(text)}</td>
<td></td>
<td>glossaries-extra v1.04+</td>
</tr>
</tbody>
</table>
Long form font used by the “user” abbreviation styles.

\texttt{\textbackslash glslowercase\{\text\}}

Converts \texttt{(text)} to lowercase.

\texttt{\textbackslash glsmakefirstuc\{\text\}}

Used by \texttt{\makefirstuc} to perform the actual case-change. As from \texttt{mfirstuc v2.08+} this just uses \texttt{\MFUsentencecase}.

\texttt{\textbackslash glsmfuaddmap\{\texttt{cs1}\}\{\texttt{cs2}\}}

If \texttt{mfirstuc v2.08+} is installed, this will use \texttt{\MFUaddmap}, otherwise it will use \texttt{\glsmfuexcl} instead. See §5.2.1 for further details.

\texttt{\textbackslash glsmfublocker\{\texttt{cs}\}}

If \texttt{mfirstuc v2.08+} is installed, this will use \texttt{\MFUblocker}, otherwise it will use \texttt{\glsmfuexcl} instead. See §5.2.1 for further details.

\texttt{\textbackslash glsmfuexcl\{\texttt{cs}\}}

If \texttt{mfirstuc v2.08+} is installed, this will use \texttt{\MFUexcl}, otherwise it will implement something similar. See §5.2.1 for further details.

\texttt{\textbackslash GLSname [\{options\}\{\texttt{entry-label}\}\{\texttt{insert}\}]} modifiers: * + \{alt-mod\}

As \texttt{\glsname} but converts the link text to all caps. This command is incompatible with some abbreviation styles.
Command Summary

\Glsname\[(options)\]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩ glossaries

As \glsname but converts the link text to sentence case. Use \Glossentryname within custom glossary styles instead of this command.

\glstext\[(options)\]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩ glossaries

References the entry identified by ⟨entry-label⟩. The text produced is obtained from the name value. The ⟨insert⟩ argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \glslink options. Use \glossentryname within custom glossary styles instead of this command.

\glsnameaccessdisplay\{(text)\}{⟨entry-label⟩} glossaries-accsupp

Does ⟨text⟩ with the access replacement text (if set).

\glsnamefont\{(text)\} glossaries

Used by \glossentryname to apply a font change to the name, unless (with glossaries-extra) the glossnamefont attribute has been set.

\glsnextpages glossaries v3.0+

Designed for use with makeindex and xindy, this may be placed in an entry’s description to override nonumberlist.

\glsnoexpandfields glossaries v3.08a+

Don’t expand field values when defining entries, except for those that explicitly have expansion enabled with \glssetexpandfield.

\glsnoidxdisplayloc\{(prefix)\}{⟨counter⟩}{⟨format⟩}{⟨location⟩} glossaries v4.04+

§11.5.6; 556

751
Command Summary

Used to display a location in the location list.

\glsnonextpages  
glossaries v1.17+

Designed for use with makeindex and xindy, this may be placed in an entry’s description to suppress the entry’s location list.

\glsnumberformat{⟨location(s)⟩}  
glossaries

The default format for entry locations. If hyperlinks are defined, this will use \glshyper-number otherwise it will simply display its argument, which may be a single location, or locations delimited by \delimR or \delimN.

\glsnumbersgroupname \ initial: Numbers  
glossaries (language-sensitive)

Expands to the title of the numbers group and (if the numbers package option is used) the numbers glossary.

\glspagelistwidth  
glossary-long & glossary-super

A length register used to set the width of the location list column for tabular-like styles.

\glspar  
glossaries

Paragraph break (for instances where \par can’t be used directly).

\glspdffmtfull{⟨entry-label⟩}  
glossaries-extra v1.42+ §5.3.2; 210

Shortcut for \glsentrylong{⟨entry-label⟩} (\glsentryshort{⟨entry-label⟩}) for use in PDF bookmarks or other text-only contexts.
Command Summary

\glspdffmtfullpl{⟨entry-label⟩}  
 Shortcut for \glsentrylongpl{⟨entry-label⟩} (\glsentryshortpl{⟨entry-label⟩}) for use in PDF bookmarks or other text-only contexts.

\glspcntchar  
 Expands to a literal percent sign.

\GLSpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]  
 modifiers: * + ⟨alt-mod⟩  
 As \glsp but converts the link text to all caps.

\GLsp[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]  
 modifiers: * + ⟨alt-mod⟩  
 As \glsp but converts the first character of the link text to uppercase (for the start of a sentence) using \makefirstuc.

\glsp[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]  
 modifiers: * + ⟨alt-mod⟩  
 As \glsp but uses the relevant plural form.

\GLSplural[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]  
 modifiers: * + ⟨alt-mod⟩  
 As \glspplural but converts the link text to all caps. If you have defined the entry with \newabbreviation use \GLSxtrshortpl or \GLSpl[preunset] instead.
As \texttt{\glsplural} but converts the first character of the link text to uppercase (for the start of a sentence) using \texttt{\makefirstuc}. If you have defined the entry with \texttt{\newabbreviation} use \texttt{\Glsxtrshortpl} or \texttt{\Glspl[preunset]} instead.

\begin{verbatim}
\glsplural[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
glossaries
\end{verbatim}

References the entry identified by \texttt{⟨entry-label⟩}. The text produced is obtained from the \texttt{plural} value. The \texttt{⟨insert⟩} argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. If you have defined the entry with \texttt{\newabbreviation} use \texttt{\glsxtrshortpl} for the short form or \texttt{\gls[preunset]}, as some abbreviation styles are too complicated to work with \texttt{\glsplural}. For the first optional argument, see \texttt{\glslink options}.

\begin{verbatim}
\glspluralaccessdisplay{⟨text⟩}{⟨entry-label⟩} glossaries-accsupp
\end{verbatim}

Does \texttt{⟨text⟩} with the \texttt{pluralaccess} replacement text (if set).

\begin{verbatim}
\glspluralsuffix
glossaries
\end{verbatim}

Expands to the letter “s” and is used to form default plurals. This command isn’t language-sensitive as there’s guarantee when it will be expanded. (It may be expanded when the entry is defined or it may be expanded when the entry is used). If you need to suppress this suffix for abbreviations, use the \texttt{noshortplural} attribute. If you need an apostrophe before the “s” for single-letter abbreviations to avoid ambiguity, use the \texttt{aposplural} attribute.

\begin{verbatim}
\glspostdescription glossaries
\end{verbatim}

A hook that is usually placed after the description in glossary styles. Some of the styles provided with the glossaries package don’t use this hook. The glossaries-extra-stylemods redefines those styles to include the hook. The default definition of this command tests for the \texttt{nopostdot} option, but the \texttt{postpunc} option redefines the command to implement the chosen punctuation.

\begin{verbatim}
\glspostinline glossary-inline v3.03+
\end{verbatim}

754
Command Summary

Used at the end of the theglossary environment.

\glspostinlinedescformat{⟨description⟩}{⟨symbol⟩}{⟨location list⟩}
glossary-inline v3.03+

Formats the top-level entry’s description, symbol and location list.

\glspostinlinesubdescformat{⟨description⟩}{⟨symbol⟩}{⟨location list⟩}
glossary-inline v3.03+

Formats the child entry’s description, symbol and location list.

\glspostlinkhook
glossaries v4.16+

A post-link hook used after all the \gls-like and \glstext-like commands. This is redefined by glossaries-extra to use \glsxtrpostlinkhook.

\glsprefixsep initial: empty
glossaries-prefix v4.45+

Separator between the prefix and the term.

\glsprestandardsort{⟨sort cs⟩}{⟨type⟩}{⟨entry-label⟩}
glossaries v3.13a+

Hook used with sort=standard to adjust the default sort value (with \makeglossaries or \makenoidxglossaries only).

\glsprepostlink{⟨category⟩}{⟨code⟩}
glossaries-extra v1.49+

Prepends ⟨code⟩ to post-link hook associated with the category identified by the label ⟨category⟩ (or simply defines it, if it doesn’t already exist).

§5.5.4; 250

755
## Command Summary

### \glsps{⟨entry-label⟩}

Shortcut for \glsxtrp{short}{⟨entry-label⟩}.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsps{⟨entry-label⟩}</td>
<td>glossaries-extra v1.07+</td>
</tr>
</tbody>
</table>

### \glspt{⟨entry-label⟩}

Shortcut for \glsxtrp{text}{⟨entry-label⟩}.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glspt{⟨entry-label⟩}</td>
<td>glossaries-extra v1.07+</td>
</tr>
</tbody>
</table>

### \glsrefentry{⟨entry-label⟩}

References (using \ref) the entry counter or sub-counter (if entrycounter or subentry-counter options are set) otherwise just does \gls{⟨entry-label⟩}.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsrefentry{⟨entry-label⟩}</td>
<td>glossaries v3.0+</td>
</tr>
</tbody>
</table>

### \glsrenewcommand{⟨cs⟩}[⟨n⟩][⟨default⟩]{⟨definition⟩}

Like \renewcommand but only issues a warning instead of an error if the command hasn’t been defined.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsrenewcommand{⟨cs⟩}[⟨n⟩][⟨default⟩]{⟨definition⟩}</td>
<td>modifier: * glossaries-extra-bib2gls v1.37+</td>
</tr>
</tbody>
</table>

### \glsreset{⟨entry-label⟩}

Globally resets the entry’s first use flag. That is, this marks the entry as “not used”.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsreset{⟨entry-label⟩}</td>
<td>glossaries</td>
</tr>
</tbody>
</table>

### \glsresetall[⟨types⟩]

Globally resets all entries associated with the listed glossaries or all glossaries if ⟨types⟩ is omitted.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsresetall[⟨types⟩]</td>
<td>glossaries</td>
</tr>
</tbody>
</table>

### \glsresetcurrcountfalse

Sets \ifglsresetcurrcount to false.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsresetcurrcountfalse</td>
<td>glossaries-extra v1.49+</td>
</tr>
</tbody>
</table>
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsresetcurrcounttrue</td>
<td>Sets \ifglsresetcurrcount to true.</td>
<td>glossaries-extra v1.49+</td>
</tr>
<tr>
<td>\glsresetentrylist</td>
<td>Inserted into the glossary code to counteract the effect of \glsnonextpages.</td>
<td>glossaries</td>
</tr>
<tr>
<td>\glssee{\langle tag\rangle}{\langle entry-label\rangle}{\langle xr-list\rangle}</td>
<td>Indexes the entry identified by \langle entry-label\rangle as a general cross-reference to the entries identified in the comma-separated list \langle xr-list\rangle. The optional argument is the textual tag that’s inserted before the cross-reference list and defaults to \seename.</td>
<td>glossaries v1.17+</td>
</tr>
<tr>
<td>\glsseefirstitem{\langle entry-label\rangle}</td>
<td>Used by \glsseelist to format the first entry.</td>
<td>glossaries-extra v1.47+</td>
</tr>
<tr>
<td>\glsseeformat{\langle tag\rangle}{\langle xr-list\rangle}{\langle location\rangle}</td>
<td>Used to format the see cross-reference in the location list. This requires a location argument for makeindex even though it isn’t required. The default definition is \textit{\langle tag\rangle} \glssee{\langle xr-list\rangle}.</td>
<td>glossaries v1.17+</td>
</tr>
<tr>
<td>\glsseeitem{\langle entry-label\rangle}</td>
<td>Used by \glsseeitem to format each entry.</td>
<td>glossaries v1.17+</td>
</tr>
<tr>
<td>\glsseeitemformat{\langle entry-label\rangle}</td>
<td>Used by \glsseeitem to produce the hyperlink text.</td>
<td>glossaries v3.0+</td>
</tr>
</tbody>
</table>

---

\glsresetcurrcounttrue glossaries-extra v1.49+  
Sets \ifglsresetcurrcount to true.

\glsresetentrylist glossaries  
Inserted into the glossary code to counteract the effect of \glsnonextpages.

\glssee{\langle tag\rangle}{\langle entry-label\rangle}{\langle xr-list\rangle} glossaries v1.17+  
Indexes the entry identified by \langle entry-label\rangle as a general cross-reference to the entries identified in the comma-separated list \langle xr-list\rangle. The optional argument is the textual tag that’s inserted before the cross-reference list and defaults to \seename.

\glsseefirstitem{\langle entry-label\rangle} glossaries-extra v1.47+  
Used by \glsseelist to format the first entry.

\glsseeformat{\langle tag\rangle}{\langle xr-list\rangle}{\langle location\rangle} glossaries v1.17+  
Used to format the see cross-reference in the location list. This requires a location argument for makeindex even though it isn’t required. The default definition is \textit{\langle tag\rangle} \glssee{\langle xr-list\rangle}.

\glsseeitem{\langle entry-label\rangle} glossaries v1.17+  
Used by \glsseeitem to format each entry.

\glsseeitemformat{\langle entry-label\rangle} glossaries v3.0+  
Used by \glsseeitem to produce the hyperlink text.
Command Summary

\texttt{\textbackslash glsseelastoxfordsep} \hspace{1cm} glossaries-extra v1.47+ \hspace{1cm} §5.13; 301

Used by \texttt{\textbackslash glsseelist} as a separator between penultimate and final entry in the list if there are at least three entries in the list.

\texttt{\textbackslash glsseelastsep} \hspace{1cm} glossaries v1.17+ \hspace{1cm} §5.13; 301

Used by \texttt{\textbackslash glsseelist} as a separator between penultimate and final entry in the list.

\texttt{\textbackslash glsseelist\{\textbackslash csv-list\}\}} \hspace{1cm} glossaries v1.17+ \hspace{1cm} §5.13; 299

Iterates over a comma-separated list of entry labels \texttt{\textbackslash csv-list} and formats them. Each label in the list is encapsulated with \texttt{\textbackslash glsseeitem} (or \texttt{\textbackslash mglssseeitem}, the label corresponds to a multi-entry). The separators are \texttt{\textbackslash glsseelastsep} (between the penultimate and last items) and \texttt{\textbackslash glsseesep} (between all the other items). With glossaries-extra, the first label is encapsulated with \texttt{\textbackslash glsseeirstitem} (or \texttt{\textbackslash mglssseeirstitem}) and the final separator for a list consisting of at least three items is given by \texttt{\textbackslash glsseelastoxfordsep}.

\texttt{\textbackslash glsseesep} \hspace{1cm} initial: ,\hspace{1cm} glossaries v1.17+ \hspace{1cm} §5.13; 301

Used by \texttt{\textbackslash glsseelist} as a separator between each entry except the last pair.

\texttt{\textbackslash glssentencecase\{\textbackslash text\}\}} \hspace{1cm} glossaries v4.50+ & glossaries-extra v1.49+ \hspace{1cm} §5.2.1; 197

Used by sentence case commands, such as \texttt{\textbackslash Gls}, to perform the case change. This is simply defined to use \texttt{\textbackslash makefirstuc}.

\texttt{\textbackslash glssetabbrvfmt\{\textbackslash category\}\}} \hspace{1cm} §4.5.2; 159

Implements the \texttt{\textbackslash display definitions} code for the abbreviation style associated with the given category.
**Command Summary**

\texttt{\glssetattribute{\langle entry-label \rangle}{\langle attribute \rangle}{\langle value \rangle}}

Locally sets the given attribute to \langle value \rangle for the category associated with the entry identified by \langle entry-label \rangle.

\texttt{\glssetcategoriesattribute{\langle category list \rangle}{\langle attribute \rangle}{\langle value \rangle}}
glossaries-extra v1.48+

Globally sets the given attribute to \langle value \rangle for all the categories in the comma-separated list \langle category list \rangle.

\texttt{\glssetcategoriesattributes{\langle category list \rangle}{\langle attribute list \rangle}{\langle value \rangle}}
glossaries-extra v1.48+

Globally sets each attribute in the comma separated \langle attribute list \rangle to \langle value \rangle for each category in the comma-separated list \langle category list \rangle.

\texttt{\glssetcategoryattribute{\langle category \rangle}{\langle attribute \rangle}{\langle value \rangle}}

Locally sets the given attribute to \langle value \rangle for the given category.

\texttt{\glssetcategoryattributes{\langle category \rangle}{\langle attribute list \rangle}{\langle value \rangle}}
glossaries-extra v1.49+

Globally sets each attribute in the comma separated \langle attribute list \rangle to \langle value \rangle for the given \langle category \rangle.

\texttt{\glssetcombinedsepabbrvnbsp}
glossaries-extra v1.48+

Defines the multi-entry separators to use a non-breaking space (\textasciitilde) for abbreviations.

\texttt{\glssetcombinedsepabbrvnone}
glossaries-extra v1.48+

Defines the multi-entry separators to use no separators for abbreviations.
Command Summary

Defines the multi-entry separators to use no separator for abbreviations.

\glssetcombinedsepnarrow{\langle width\rangle}{\langle narrow-sep\rangle} glossaries-extra v1.48+

§7.4; 346

Defines the multi-entry separators to use \(\langle narrow-sep\rangle\) if the width of associated field values is less than \(\langle width\rangle\).

\glssetexpansionfield{\langle field\rangle} glossaries v3.13a+

Expand the value of the field identified by its internal field label when defining entries (overrides \glsexpandfields).

\glssetnoexpansionfield{\langle field\rangle} glossaries v3.13a+

Don’t expand the value of the field identified by its internal field label when defining entries (overrides \glsexpandfields).

\glssetregularcategory{\langle category\rangle} §10.2.2; 504

Locally sets the regular attribute to true for the given category.

\glssetwidest[\langle level\rangle]{\langle name\rangle} glossary-tree

Indicates that \(\langle name\rangle\) is the widest name for the given hierarchical level.

\glssetshortaccessdisplay{\langle text\rangle}{\langle entry-label\rangle} glossaries-accsupp

Does \(\langle text\rangle\) with the shortaccess replacement text (if set).

\glssetshortaccsupp{\langle replacement\rangle}{\langle content\rangle} glossaries-accsupp v4.45+
Applies \( replacement \) as the expansion \( content \) attribute for \( content \) using \texttt{\glsaccessibility}.

\texttt{\glsshortpltok}

A token register that stores the short plural form (which may have been modified after being passed to \texttt{\newabbreviation}).

\texttt{\glsshortpluralaccessdisplay}\{\texttt{text}\}\{\texttt{entry-label}\}\texttt{glossaries-accsupp}

Does \texttt{text} with the \texttt{shortpluralaccess} replacement text (if set).

\texttt{\glsshorttok}\texttt{glossaries}

A token register that stores the short form (which may have been modified after being passed to \texttt{\newabbreviation}).

\texttt{\glsshowtarget}\{\texttt{target-name}\}\texttt{glossaries v4.32+}

Formats the target name when \texttt{debug=showtargets} is enabled using either \texttt{\glsshowtarget-inner} or \texttt{\glsshowtargetouter}, depending on the current mode.

\texttt{\glsshowtargetfont}\texttt{glossaries v4.45+}

Font declaration used by debugging annotations.

\texttt{\glsshowtargetfonttext}\{\texttt{text}\}\texttt{glossaries v4.50+}

Text-block command that checks for math mode and switches to the font given by the \texttt{\glsshowtargetfont} declaration.
\textbf{Command Summary}

\texttt{\textbackslash glsshowtargetinner\{\textit{target-name}\}} \hspace{1cm} \textit{glossaries v4.50+}

Formats the target name for inner and maths mode when \texttt{debug=showtargets} is enabled.

\texttt{\textbackslash glsshowtargetinnersymleft\{\textit{name}\}} \hspace{1cm} \textit{glossaries-extra v1.48+} \hspace{1cm} §2.5; 30

Shows the left inner annotation followed by the left marker symbol \texttt{\textbackslash glsxtrshowtargetsymbolleft}.

\texttt{\textbackslash glsshowtargetinnersymright\{\textit{name}\}} \hspace{1cm} \textit{glossaries-extra v1.48+} \hspace{1cm} §2.5; 30

Shows the right marker symbol \texttt{\textbackslash glsxtrshowtargetsymbolright} followed by the right inner annotation.

\texttt{\textbackslash glsshowtargetouter\{\textit{target-name}\}} \hspace{1cm} \textit{glossaries v4.50+}

Formats the target name for outer mode when \texttt{debug=showtargets} is enabled. This places a marker (\texttt{\textbackslash glsshowtargetsymbol}) in the text and \texttt{\textit{target-name}} in the margin.

\texttt{\textbackslash glsshowtargetsymbol} \hspace{1cm} \textit{glossaries v4.45+}

Marker (.) used in debugging annotations.

\texttt{\textbackslash glssubentryitem\{\textit{entry-label}\}} \hspace{1cm} \textit{glossaries v3.0+}

Does nothing if \texttt{subentrycounter=false}, otherwise increments and displays the associated counter.

\texttt{\textbackslash glssubgroupheading\{\textit{previous level}\}\{\textit{level}\}\{\textit{parent-label}\}\{\textit{group-label}\}} \hspace{1cm} \textit{glossaries-extra v1.49+} \hspace{1cm} §8.4.1; 393

Used to format sub-group headings. Only applicable with the “unsrt” family of commands. This command won’t occur in glossaries that use \texttt{\textbackslash printglossary} or \texttt{\textbackslash printnoidxglossary}.


**Command Summary**

\GLSsymbol[(options)]{(entry-label)}{(insert)}\] modifiers: * + ⟨alt-mod⟩

As \glssymbol but converts the link text to all caps.

\Glssymbol[(options)]{(entry-label)}{(insert)}\] modifiers: * + ⟨alt-mod⟩

As \glssymbol but converts the link text to sentence case. Use \Glossentrysymbol within custom glossary styles instead of this command.

\glssymbol[(options)]{(entry-label)}{(insert)}\] modifiers: * + ⟨alt-mod⟩

References the entry identified by ⟨entry-label⟩. The text produced is obtained from the symbol value. The ⟨insert⟩ argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \glslink options. Use \glossentrysymbol within custom glossary styles instead of this command.

\glssymbolaccessdisplay{(text)}{(entry-label)} glossaries-accsupp

Does ⟨text⟩ with the symbolaccess replacement text (if set).

\glssymbolplural[(options)]{(entry-label)}{(insert)} modifiers: * + ⟨alt-mod⟩

As \glssymbol but for the symbolplural field.

\glssymbolpluralaccessdisplay{(text)}{(entry-label)} glossaries-accsupp

Does ⟨text⟩ with the symbolpluralaccess replacement text (if set).
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glossarygrouplname</code></td>
<td>Initial: Symbols glossaries (language-sensitive)</td>
<td>§8.7.4; 469</td>
</tr>
<tr>
<td><code>\glstabledescheader</code></td>
<td>Header for the description column.</td>
<td>§8.7.4; 469</td>
</tr>
<tr>
<td><code>\glstableiffilter{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}</code></td>
<td>Internally used by the custom handler in <code>\printunrstatetable</code> to perform additional filtering.</td>
<td>§8.7.4; 468</td>
</tr>
<tr>
<td><code>\glstablenameheader</code></td>
<td>Header for the name column.</td>
<td>§8.7.4; 469</td>
</tr>
<tr>
<td><code>\glstableotherheader</code></td>
<td>Header for the other column.</td>
<td>§8.7.4; 470</td>
</tr>
<tr>
<td><code>\glstablenumberheader</code></td>
<td>Header for the symbol column.</td>
<td>§8.7.4; 469</td>
</tr>
<tr>
<td><code>\glossaryglossary</code></td>
<td>Used by glossary styles to create a hypertext (if enabled) for the entry (identified by <code>⟨entry-label⟩</code>). The <code>⟨text⟩</code> is usually <code>\glossaryglossaryname{⟨entry-label⟩}</code>, but it can be something else.</td>
<td>§8.7.4; 469</td>
</tr>
</tbody>
</table>

Expands to the title of the `symbols` group and (if the `symbols` package option is used) the `symbols` glossary.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\GLStext\⟨options⟩\{⟨entry-label⟩\}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩ glossaries</code></td>
<td>As <code>\glstext</code> but converts the link text to all caps. If you have defined the entry with <code>\newabbreviation</code> use <code>\GLSxtrshort</code> or <code>\GLS[preunset]</code> instead.</td>
</tr>
<tr>
<td><code>\Glstext\⟨options⟩\{⟨entry-label⟩\}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩ glossaries</code></td>
<td>As <code>\glstext</code> but converts the first character of the link text to uppercase (for the start of a sentence) using <code>\makefirstuc</code>. If you have defined the entry with <code>\newabbreviation</code> use <code>\Glsxtrshort</code> or <code>\Gls[preunset]</code> instead.</td>
</tr>
<tr>
<td><code>\glstext\⟨options⟩\{⟨entry-label⟩\}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩ glossaries</code></td>
<td>References the entry identified by <code>⟨entry-label⟩</code>. The text produced is obtained from the <code>text</code> value. The <code>⟨insert⟩</code> argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. If you have defined the entry with <code>\newabbreviation</code> use <code>\glstext</code> for the short form or <code>\Gls[preunset]</code>, as some abbreviation styles are too complicated to work with <code>\glstext</code>. For the first optional argument, see <code>\glslink</code> options.</td>
</tr>
<tr>
<td><code>\glstextaccessdisplay\{⟨text⟩\}{⟨entry-label⟩}</code> glossaries-accsupp</td>
<td>Does <code>⟨text⟩</code> with the <code>textaccess</code> replacement text (if set).</td>
</tr>
<tr>
<td><code>\glstextformat\{⟨text⟩\}</code> glossaries v1.04+</td>
<td>The default outer text formatting command used by the <code>\gls</code>-like and <code>\glstext</code>-like commands.</td>
</tr>
<tr>
<td><code>\glstextup\{⟨text⟩\}</code> glossaries v3.09a+</td>
<td>Counteracts the effect of <code>\textsc</code>.</td>
</tr>
</tbody>
</table>
Command Summary

\texttt{\textbackslash glstildechar}  
\textit{glossaries v4.10+}

Expands to a literal tilde character.

\texttt{\textbackslash glstopicAssignSubIndent\{\langle level\rangle\}}  
\textit{glossary-topic v1.40+} \§8.7.3; 467

Used to set the indentation for sub-levels.

\texttt{\textbackslash glstopicAssignWidest\{\langle level\rangle\}}  
\textit{glossary-topic v1.40+} \§8.7.3; 467

Used by \texttt{\textbackslash glstopicAssignSubIndent} to calculate the width of the widest name for the given level.

\texttt{\textbackslash glstopicCols}  
\textit{glossary-topic v1.40+} \§8.7.3; 462

Expands to the number of columns for topicmcols.

\texttt{\textbackslash glstopicColsEnv}  
\textit{glossary-topic v1.40+} \§8.7.3; 462

Expands to the multicols environment name to use for topicmcols.

\texttt{\textbackslash glstopicDesc\{\langle entry-label\rangle\}}  
\textit{glossary-topic v1.40+} \§8.7.3; 466

Used to format the top-level description.

\texttt{\textbackslash glstopicGroupHeading\{\langle group-label\rangle\}}  
\textit{glossary-topic v1.40+} \§8.7.3; 465

Used to format the top-level group headings, if required.
Command Summary

Initialisation hook.

\texttt{\glstopicItem{\langle entry-label\rangle}{\langle location list\rangle}}

Used to format top-level entries.

\texttt{\glstopicLoc{\langle entry-label\rangle}{\langle location list\rangle}}

Used to format the top-level location list.

\texttt{\glstopicMarker{\langle entry-label\rangle}}

Hook inserted before a top-level entry.

\texttt{\glstopicMidSkip}

Vertical space inserted before the description for a top-level entry.

\texttt{\glstopicParIndent}

Length register used for the top-level paragraph indent.

\texttt{\glstopicPostSkip}

Vertical space inserted after the description for a top-level entry.

\texttt{\glstopicPreSkip}

Vertical space inserted before a top-level entry.
Command Summary

\glistopicSubGroupHeading{(prev group level)}{(group level)}{(parent entry)}{(group-label)}

Used to format the sub-group headings, if supported.

\glistopicSubIndent
glossary-topic v1.40+

Length register used for the child indent.

\glistopicSubItem{(level)}{(entry-label)}{(location list)}

glossary-topic v1.40+

Used to format child entries.

\glistopicSubItemBox{(level)}{(text)}
glossary-topic v1.40+

Horizontal box used for child name if a widest name has been provided.

\glistopicSubItemParIndent
glossary-topic v1.46+

Length register used for the child paragraph indent.

\glistopicSubItemSep
glossary-topic v1.40+

Horizontal separator used after child names.

\glistopicSubLoc{(entry-label)}{(location list)}
glossary-topic v1.41+

Formats the child location lists.

\glistopicSubNameFont{(text)}
glossary-topic v1.40+

768
Command Summary

Font command to apply to the child entry name.

\glstopicSubPreLocSep \hspace{1em} glossary-topic v1.41+

§8.7.3; 468

Separator before the child location lists.

\glstopicTitle\{\langle entry-label\rangle\} \hspace{1em} glossary-topic v1.40+

§8.7.3; 466

Used to format the name and (if provided) symbol for the top-level entry title.

\glstopicTitleFont\{\langle text\rangle\} \hspace{1em} glossary-topic v1.40+

§8.7.3; 466

Font command to apply to the top-level entry title.

\glstreechilddesc\{\langle entry-label\rangle\} \hspace{1em} glossaries-extra-stylemods v1.31+

§8.6.5.4; 433

Displays the given child entry’s description with pre and post hooks for the tree styles.

\glstreeChildDescLoc\{\langle entry-label\rangle\}\{\langle location list\rangle\} \hspace{1em} glossaries-extra-stylemods v1.41+

§8.6.5.4; 433

Formats the child description (if set) and location list for the tree styles.

\glstreechildpredesc \hspace{1em} glossary-tree v4.26+

§8.6.5.4; 431

Space inserted before child descriptions.

\glstreechildprelocation \hspace{1em} initial: \glstreeprelocation

\hspace{1em} glossaries-extra-stylemods v1.21+

§8.6.5.4; 432

Used before the child entry location list for the tree and index styles.

769
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glstreechildsymbol{⟨entry-label⟩}</code></td>
<td>Displays the top-level symbol in parentheses, if set, for the tree styles.</td>
</tr>
<tr>
<td><code>\glstreedefaultnamefmt{⟨text⟩}</code></td>
<td>Used as the default name format for the tree and index styles.</td>
</tr>
<tr>
<td><code>\glstreedesc{⟨entry-label⟩}</code></td>
<td>Displays the given top-level entry’s description with pre and post hooks for the tree styles.</td>
</tr>
<tr>
<td><code>\glstreeDescLoc{⟨entry-label⟩}{⟨location list⟩}</code></td>
<td>Formats the top-level description (if set) and location list for the tree styles.</td>
</tr>
<tr>
<td><code>\glstreegroupheaderfmt{⟨text⟩}</code></td>
<td>Used to format the group title for the treegroup and indexgroup styles.</td>
</tr>
<tr>
<td><code>\glstreegroupheaderskip</code></td>
<td>After group header skip for the treegroup and indexgroup styles.</td>
</tr>
<tr>
<td><code>\glstreeitem</code></td>
<td>Group skip for the tree and index styles.</td>
</tr>
</tbody>
</table>
Command Summary

Used to indent the top-level entries for the index styles.

\glstreenamefmt\{⟨text⟩\}  
glossary–tree v4.08+

Used to format the name for the tree and index styles.

\glstreenavigationfmt\{⟨text⟩\}  
glossary–tree v4.22+

§8.6.5.4; 430

Used to format the navigation element for styles like treehypergroup.

\glstreeNoDescSymbolPreLocation  
glossaries–extra–stylemods v1.42+

§8.6.5.4; 433

Inserted before the location list when there’s no description or symbol for the tree styles.

\glstreenonamechilddesc\{⟨entry-label⟩\}  
glossaries–extra–stylemods v1.31+

§8.6.5.4; 432

Displays the given child entry’s description and post hook for the treenoname styles.

\glstreenonameChildDescLoc\{⟨entry-label⟩\}  
glossaries–extra–stylemods v1.45+

§8.6.5.4; 433

Displays the given child entry’s description and location list for the treenoname styles.

\glstreenamedesc\{⟨entry-label⟩\}  
glossaries–extra–stylemods v1.31+

§8.6.5.4; 432

Displays the given top-level entry’s description with pre and post hooks for the treenoname styles.

\glstreenamedescLoc\{⟨entry-label⟩\}  
glossaries–extra–stylemods v1.45+

§8.6.5.4; 432

Displays the given top-level entry’s description and location list for the treenoname styles.

771
Command Summary

\glistreenonamesymbol{⟨entry-label⟩}  
\textit{glossaries-extra-stylemods v1.31+}  
§8.6.5.4; 432

Displays the given top-level entry’s symbol in parentheses for the treenoname styles.

\glistreepredesc  
\textit{glossary-tree v4.26+}  
§8.6.5.4; 431

Space inserted before top-level descriptions.

\glistreePreHeader{⟨group-label⟩}{⟨group-title⟩}  
\textit{glossaries-extra-stylemods v1.41+}  
§8.6.5.4; 431

Pre group header hook the treegroup and indexgroup styles.

\glistreeprelocation \textit{initial: glsxtreprelocation}  
\textit{glossaries-extra-stylemods v1.21+}  
§8.6.5.4; 432

Used before the top-level entry location list for the tree and index styles.

\glistreesubgroupitem{previous group level}{level}{parent label}{group label}{group title}  
\textit{glossaries-extra-stylemods v1.49+}  

Used to display the sub-group header in the treegroup styles.

\glistreesubitem  
\textit{glossary-tree v4.26+}  

Used to indent the level 1 entries for the index styles.

\glistreesubpreheader{⟨previous group level⟩}{⟨level⟩}{⟨parent label⟩}{⟨group label⟩}{⟨group title⟩}  
\textit{glossaries-extra-stylemods v1.49+}  

Pre sub-group header hook the treegroup and indexgroup styles.
Command Summary

\glistreesubsubitem \textit{glossary-tree} v4.26+

Used to indent the level 2 entries for the index styles.

\glistreesymbol{⟨entry-label⟩} \textit{glossaries-extra-stylemods} v1.31+

Displays the top-level symbol in parentheses, if set, for the tree styles.

\glstriggerrecordformat{⟨location⟩} \textit{glossaries-extra} v1.21+

Used as a special location format that indicates that the record is a trigger record.

\glsunexpandedfieldvalue{⟨entry-label⟩}{⟨field-label⟩} \textit{glossaries} v4.48+

For use in expandable contexts where the field value is required but the contents should not be expanded. The field should be identified by its internal field label. Expands to nothing with no error or warning if the entry or field aren’t defined.

\glsunset{⟨entry-label⟩} \textit{glossaries}

Globally unsets the entry’s first use flag. That is, this marks the entry as “used”.

\glsunsetall{⟨types⟩} \textit{glossaries}

Globally unsets all entries associated with the listed glossaries or all glossaries if ⟨types⟩ is omitted.

\glsunsetcategoryattribute{⟨category⟩}{⟨attribute⟩} \textit{glossaries-extra} v1.47+

Locally unsets the given attribute for the given category.

773
Command Summary

\glsupdatewidest{⟨level⟩}{⟨name⟩}
glossaries-extra-stylemods v1.23+

Similar to \glssetwidest but only if ⟨name⟩ is wider than the current widest value for the given hierarchical level.

\glsuppercase{⟨text⟩}
glossaries v4.50+

Converts ⟨text⟩ to uppercase.

\glsuseabbrvfont{⟨style-name⟩}{⟨text⟩}
glossaries-extra v1.21+

Formats ⟨text⟩ according to the short format for the given abbreviation style.

\glsuselongfont{⟨style-name⟩}{⟨text⟩}
glossaries-extra v1.21+

Formats ⟨text⟩ according to the long format for the given abbreviation style.

\glsuserdescription{⟨text⟩}{⟨entry-label⟩}
glossaries-extra v1.30+

Description encapsulator for styles like long-short-user.

\GLSuseri{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
glossaries v2.04+

As \glsuseri but converts the link text to all caps.

\Glsuseri{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩
glossaries v2.04+

As \glsuseri but converts the link text to sentence case.
References the entry identified by \langle entry-label\rangle. The text produced is obtained from the \texttt{user1} value. The \langle insert\rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \texttt{\glsslink} options.

\texttt{\glsuseri}\langle options\rangle \langle entry-label\rangle \langle insert\rangle\hspace{1cm} \texttt{modifiers: \^ \langle alt-mod\rangle}\hspace{1cm} \texttt{glossaries v2.04+}

\texttt{\glsuseriaccessdisplay}\langle text\rangle \langle entry-label\rangle\hspace{1cm} \texttt{glossaries-accsupp v4.45+}

Does \langle text\rangle with the \texttt{user1access} replacement text (if set).

\texttt{\GLSuserii}\langle options\rangle \langle entry-label\rangle \langle insert\rangle\hspace{1cm} \texttt{modifiers: \^ \langle alt-mod\rangle}\hspace{1cm} \texttt{glossaries v2.04+}

As \texttt{\glsuserii} but converts the link text to all caps.

\texttt{\Glsuserii}\langle options\rangle \langle entry-label\rangle \langle insert\rangle\hspace{1cm} \texttt{modifiers: \^ \langle alt-mod\rangle}\hspace{1cm} \texttt{glossaries v2.04+}

As \texttt{\glsuserii} but converts the link text to sentence case.

\texttt{\glssuserii}\langle options\rangle \langle entry-label\rangle \langle insert\rangle\hspace{1cm} \texttt{modifiers: \^ \langle alt-mod\rangle}\hspace{1cm} \texttt{glossaries v2.04+}

References the entry identified by \langle entry-label\rangle. The text produced is obtained from the \texttt{user2} value. The \langle insert\rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \texttt{\glsslink} options.

\texttt{\glsuseriaccessdisplay}\langle text\rangle \langle entry-label\rangle\hspace{1cm} \texttt{glossaries-accsupp v4.45+}

Does \langle text\rangle with the \texttt{user2access} replacement text (if set).
Command Summary

\GLSuserii ([options])\{⟨entry-label⟩\} ([insert])  modifiers: * + ⟨alt-mod⟩
glossaries v2.04+

As \glsuserii but converts the link text to all caps.

\Gluserii ([options])\{⟨entry-label⟩\} ([insert])  modifiers: * + ⟨alt-mod⟩
glossaries v2.04+

As \gluserii but converts the link text to sentence case.

\gluserii ([options])\{⟨entry-label⟩\} ([insert])  modifiers: * + ⟨alt-mod⟩
glossaries v2.04+

References the entry identified by ⟨entry-label⟩. The text produced is obtained from the user3 value. The ⟨insert⟩ argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \glslink options.

\gluseriiaccessdisplay\{⟨text⟩\} ⟨⟨entry-label⟩⟩  glossaries-accsupp v4.45+

Does ⟨text⟩ with the user3access replacement text (if set).

\GLSuseriv ([options])\{⟨entry-label⟩\} ([insert])  modifiers: * + ⟨alt-mod⟩
glossaries v2.04+

As \glsuseriv but converts the link text to all caps.

\Gluseriv ([options])\{⟨entry-label⟩\} ([insert])  modifiers: * + ⟨alt-mod⟩
glossaries v2.04+

As \gluseriv but converts the link text to sentence case.
Command Summary

\texttt{\glsuseriv[\{options\}]{\langle entry-label \rangle}}\{\langle insert \rangle\} \quad \text{modifiers: } * + \langle alt-mod \rangle

Referencing the entry identified by \langle entry-label \rangle. The text produced is obtained from the \texttt{user4} value. The \langle insert \rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \texttt{\glslink options}.

\texttt{\glsuserivaccessdisplay\{\langle text \rangle\}{\langle entry-label \rangle}}

\texttt{\Glsuserv[\{options\}]{\langle entry-label \rangle}}\{\langle insert \rangle\} \quad \text{modifiers: } * + \langle alt-mod \rangle

As \texttt{\glsuserv} but converts the link text to all caps.

\texttt{\Gluserv[\{options\}]{\langle entry-label \rangle}}\{\langle insert \rangle\} \quad \text{modifiers: } * + \langle alt-mod \rangle

As \texttt{\gluserv} but converts the link text to sentence case.

\texttt{\glsuserv[\{options\}]{\langle entry-label \rangle}}\{\langle insert \rangle\} \quad \text{modifiers: } * + \langle alt-mod \rangle

Referencing the entry identified by \langle entry-label \rangle. The text produced is obtained from the \texttt{user5} value. The \langle insert \rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \texttt{\glslink options}.

\texttt{\glsuservaccessdisplay\{\langle text \rangle\}{\langle entry-label \rangle}}

\texttt{\glsuserv[\{options\}]{\langle entry-label \rangle}}\{\langle insert \rangle\} \quad \text{modifiers: } * + \langle alt-mod \rangle

Does \langle text \rangle with the \texttt{user5access} replacement text (if set).
\GLSuservi\[\langle\text{options}\rangle\}\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}\]

As \glssuservi but converts the link text to all caps.

\Glsuservi\[\langle\text{options}\rangle\]\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}\]

As \glssuservi but converts the link text to sentence case.

\glsuservi\[\langle\text{options}\rangle\]\{\langle\text{entry-label}\rangle\}\{\langle\text{insert}\rangle\}\]

References the entry identified by \langle\text{entry-label}\rangle. The text produced is obtained from the \texttt{user6} value. The \langle\text{insert}\rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \glsslink options.

\glsuserviaccessdisplay\{\langle\text{text}\rangle\}\{\langle\text{entry-label}\rangle\}\]

Does \langle\text{text}\rangle with the \texttt{user6access} replacement text (if set).

\glswriteentry\{\langle\text{entry-label}\rangle\}\{\langle\text{code}\rangle\}\]

Performs the indexing code unless indexing should be suppressed.

\glsxpabbrvfont\{\langle\text{text}\rangle\}\{\langle\text{category}\rangle\}\]

If the \texttt{markshortwords} attribute is set for the given category, this encapsulates \langle\text{text}\rangle with \glsabbrvfont otherwise with \glsinnerfmtabbrvfont. This command has to expand, so protect any content that shouldn’t expand.
If the markwords attribute is set for the given category, this encapsulates \text with \glslongfont otherwise with \glsinnerfmtlongfont. This command has to expand, so protect any content that shouldn’t expand.

\textbf{Glsxstr}

$$\texttt{\Glsxtr[(gls-options)][(dfn-options)]\{entry-label\}}$$

As \glsxstr but applies sentence case.

$$\texttt{\Glsxtr[(gls-options)][(dfn-options)]\{entry-label\}}$$

If \texttt{entry-label} has already been defined, this just references it, otherwise the entry is defined. This command must be enabled with \GlsXtrEnableOnTheFly.

$$\texttt{\glsxtr@counterrecord\{entry-label\}\{counter\}\{value\}}$$

An aux file command used with \GlsXtrRecordCounter to append \texttt{value} to the record. \texttt{counter} field. Also implements \glsxtrAddCounterRecordHook.

$$\texttt{\glsxtr@record\{entry-label\}\{h-prefix\}\{counter\}\{encap\}\{location\}}$$

Used in the aux file to provide the record for \texttt{bib2gls} (record=only). Ignored by \LaTeX.

$$\texttt{\glsxtr@record@nameref\{entry-label\}\{h-prefix\}\{counter\}\{encap\}\{location\}\{current title\}\{current anchor\}\{the-h-counter\}}$$

Used in the aux file to provide the nameref record for \texttt{bib2gls}. Ignored by \LaTeX.
Command Summary

\glsxtr@resource\{⟨options⟩\}\{⟨basename⟩\}  
\textit{glossaries-extra v1.08+}  
§11; 511

Used in the aux file to provide the resource options for \textit{bib2gls} for each resource set. Ignored by \texttt{\LaTeX}.

\glsxtrabbreviationfont\{⟨text⟩\}  
\textit{glossaries-extra v1.30+}  
§5.5.2; 240

Used by \texttt{\glsentryfmt} to encapsulate non-regular entries the have the \texttt{short} field set.

\glsxtrabbrvfootnote\{⟨entry-label⟩\}\{⟨text⟩\}  
\textit{glossaries-extra v1.07+}  
145

Command that produces the footnote for the footnote abbreviation styles, such as \texttt{short-footnote} and \texttt{short-postfootnote}.

\glsxtrabbrvpluralsuffix  \textit{initial: \texttt{\glspluralsuffix}}  
\textit{glossaries-extra v1.12+}  
§4.1.2; 44

The default plural suffix used for abbreviations.

\glsxtrabbrvtype  \textit{initial: \texttt{\glsdefaulttype}}  
\textit{glossaries-extra}  
§4.1.4; 44

Expands to the label of the default abbreviation glossary. The \textit{abbreviations} package option will redefine this to \texttt{abbreviations}.

\glsxtrAccSuppAbbrSetFirstLongAttrs\{⟨category⟩\}  
\textit{glossaries-extra v1.42+}  
§4.5.3.1; 169

Initialised accessibility support for the \texttt{name}, \texttt{text} and \texttt{plural} fields (if enabled with \texttt{accsupp}). This command is provided for abbreviation styles where the \texttt{name} and \texttt{text} are just the formatted abbreviation.

\glsxtrAccSuppAbbrSetNameLongAttrs\{⟨category⟩\}  
\textit{glossaries-extra v1.42+}  
§4.5.3.1; 169

780
Initialised accessibility support for the \texttt{first}, \texttt{firstplural}, \texttt{text} and \texttt{plural} fields (if enabled with \texttt{accsupp}). This command is provided for abbreviation styles where the \texttt{first} and \texttt{text} are just the formatted abbreviation.

\begin{verbatim}
\glsxtrAccSuppAbbrSetNameShortAttrs{\langle category\rangle}
glossaries-extra v1.42+
\end{verbatim}

§4.5.3.1; 169

Initialised accessibility support for the \texttt{name} field (if enabled with \texttt{accsupp}). This command is provided for abbreviation styles where only the \texttt{name} is just the formatted abbreviation.

\begin{verbatim}
\glsxtrAccSuppAbbrSetNoLongAttrs{\langle category\rangle}
glossaries-extra v1.42+
\end{verbatim}

§4.5.3.1; 168

Initialised accessibility support for the \texttt{name}, \texttt{first}, \texttt{firstplural}, \texttt{text} and \texttt{plural} fields (if enabled with \texttt{accsupp}). This command is provided for abbreviation styles where the \texttt{name}, \texttt{first} and \texttt{text} are just the formatted abbreviation.

\begin{verbatim}
\glsxtrAccSuppAbbrSetTextShortAttrs{\langle category\rangle}
glossaries-extra v1.42+
\end{verbatim}

§4.5.3.1; 169

Initialised accessibility support for the \texttt{text} and \texttt{plural} fields (if enabled with \texttt{accsupp}). This command is provided for abbreviation styles where the \texttt{text} is just the formatted abbreviation.

\begin{verbatim}
\glsxtractivatenopost
glossaries-extra v1.22+
\end{verbatim}

§8.5; 415

Activates \texttt{\nopostdesc} and \texttt{\glsxtrnopostpunc}.

\begin{verbatim}
\glsxtractualanchor
glossaries-extra-bib2gls v1.49+
\end{verbatim}

§11.5.6; 557

Expands to the anchor required by \texttt{\glsxtrdisplaylocnameref}.

\begin{verbatim}
\glsxtraddallcrossrefs
\end{verbatim}

§5.9.3; 279

Iterates over all defined entries and indexes any cross-references (identified by the \texttt{see} or \texttt{seealso} keys) that haven’t been used.
\glsxtrAddCounterRecordHook{⟨entry-label⟩}{⟨counter⟩}{⟨value⟩}
glossaries-extra v1.49+

User-level hook used by \glsxtr@counterrecord. If this command is redefined, it must be done so in the preamble before the aux file is input.

\glsxtraddgroup{⟨entry-label⟩}{⟨code⟩}
glossaries-extra v1.49+

Used by the “unsrt” family of commands to perform ⟨code⟩ if the entry identified by ⟨entry-label⟩ should have support for groups.

\glsxtraddlabelprefix{⟨label-prefix⟩}
glossaries-extra-bib2gls v1.37+

Appends ⟨label-prefix⟩ to the list of known labels.

\glsxtraddpunctuationmark{⟨token(s)⟩}

Adds ⟨token(s)⟩ to the list of punctuation characters used by \glsxtrifnextpunc. You may list multiple characters at the same time to add a batch, but don’t add any separators (including spaces). Note that each character must be a single token, which means a single-byte character for pdf\TeX. Multi-byte characters (UTF-8) will required a native Unicode engine (X\LaTeX or Lua\LaTeX).

\glsxtraddunusedxrefs
glossaries-extra v1.49+

Indexes any cross-references (identified by the see or seealso keys) that haven’t been used.

\glsxtralias{⟨entry-label⟩}
glossaries-extra v1.12+

Expands to the value of the alias field for the entry identified by ⟨entry-label⟩. If the field isn’t set, this will expand to nothing. If the entry isn’t defined, this will expand to \relax.
Hook implemented when the alias key is provided when an entry is defined.

Length register for the subsequent paragraph indentation for the alttree-like styles.

Initialisation code performed by the alttree-like styles.

Formats the symbol, description and location for child entries for the alttree-like styles.

Formats the symbol, description and location for top-level entries for the alttree-like styles.

For use with fields that should contain comma-separated lists, this will append a command followed by \langle element\rangle to the field value. If the field isn’t defined, this command will behave like \glsxtrdeffield. No existence check is performed.

Locally append \langle options\rangle to the default options for the \glslike and \glstext-like commands.
## Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Required Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrassignactualsetup \text{\texttt{(requires accsupp)}}</td>
<td>Used to strip common formatting commands from a field value to supply the text-only accessibility content when initialising the default \texttt{shortaccess} and \texttt{shortpluralaccess} values.</td>
<td>§9.1; 475</td>
</tr>
<tr>
<td>\glsxtrassignfieldfont \text{\texttt{{entry-label}}}</td>
<td>Used by the \texttt{\glstext}-like commands to initialise the formatting commands required for the given entry.</td>
<td>§5.5.2; 240</td>
</tr>
<tr>
<td>\glsxtrassignlinktextfmt</td>
<td>Initialised by the \texttt{\gls}-like and \texttt{\glstext}-like commands, this contains the definitions of \texttt{\glslabel}, \texttt{\glstextformat}, \texttt{\glsxtrgenentrytextfmt}.</td>
<td>§5.5.4; 254</td>
</tr>
<tr>
<td>\glsxtrattrentrytextfmt \text{\texttt{{text}}}</td>
<td>Applies the command obtained from the control sequence name supplied in the \texttt{innertext-format} attribute for the category assigned to the entry given by \texttt{\glslabel}. This command isn’t used by default as it should rarely be needed an increases complexity.</td>
<td>§5.5.3; 243</td>
</tr>
<tr>
<td>\GlsXtrAutoAddOnFormat \text{\texttt{{entry-label}{format list}{glsadd options}}}</td>
<td>Identifies formats that should trigger an automatic \texttt{\glsadd} by the \texttt{\gls}-like and \texttt{\glstext}-like commands.</td>
<td>§5.8; 260</td>
</tr>
<tr>
<td>\glsxtrautoindex \text{\texttt{\textit{initial: \index}}}</td>
<td>The indexing command used by by the auto-indexing feature.</td>
<td>§12; 575</td>
</tr>
<tr>
<td>\glsxtrautoindexassignsort \text{\texttt{{entry-label}}}</td>
<td></td>
<td>§12; 575</td>
</tr>
</tbody>
</table>
Command Summary

Used to assign the sort value for the auto-indexing feature.

\glsxtrautoindexentry{⟨entry-label⟩}  \hfill glossaries-extra v1.16+

Expands to the “actual” part for the auto-indexing feature.

\glsxtrautoindexesc  \hfill glossaries-extra v1.36+

Escapes the sort value used by the auto-indexing feature.

\glsxtrBasicDigitrules  \hfill glossaries-extra-bib2gls v1.27+

Expands to the Basic Latin digit character sort rules.

\glsxtrbibaddress{⟨entry-label⟩}  \hfill (defined by \GlsXtrProvideBibTeXFields) §11.5.2; 536

Accesses the \textit{address} field.

\glsxtrbibauthor{⟨entry-label⟩}  \hfill (defined by \GlsXtrProvideBibTeXFields) §11.5.2; 536

Accesses the \textit{author} field.

\glsxtrbibbooktitle{⟨entry-label⟩}  \hfill (defined by \GlsXtrProvideBibTeXFields) §11.5.2; 536

Accesses the \textit{booktitle} field.

\glsxtrbibchapter{⟨entry-label⟩}  \hfill (defined by \GlsXtrProvideBibTeXFields) §11.5.2; 536

Accesses the \textit{chapter} field.

785
Command Summary

\texttt{\textbackslash glsxtbibedition\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{edition} field.

\texttt{\textbackslash glsxtbibhowpublished\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{howpublished} field.

\texttt{\textbackslash glsxtbibinstitution\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{institution} field.

\texttt{\textbackslash glsxtbibjournal\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{journal} field.

\texttt{\textbackslash glsxtbibmonth\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{month} field.

\texttt{\textbackslash glsxtbibnote\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{note} field.

\texttt{\textbackslash glsxtbibnumber\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{number} field.

\texttt{\textbackslash glsxtbiborganization\{\textit{entry-label}\}} \quad (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the \textit{organization} field.
Command Summary

Accesses the organization field.

\texttt{\textbackslash gluextrbibpages\{(entry-label)\}} (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the pages field.

\texttt{\textbackslash gluextrbibpublisher\{(entry-label)\}} (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the publisher field.

\texttt{\textbackslash gluextrbibschool\{(entry-label)\}} (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the school field.

\texttt{\textbackslash gluextrbibseries\{(entry-label)\}} (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the series field.

\texttt{\textbackslash \texttt{GlsXtrBibTeXEntryAliases}} glossaries-extra-bib2gls v1.29+

Expands to the \LaTeX{} to bib2gls entry aliases for use in \texttt{\textbackslash entry-type-aliases}.

\texttt{\textbackslash gluextrbibtitle\{(entry-label)\}} (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the title field.

\texttt{\textbackslash gluextrbibtype\{(entry-label)\}} (defined by \texttt{\textbackslash GlsXtrProvideBibTeXFields})

Accesses the bibtextype field.
Command Summary

\texttt{\textbackslash glsxtrbibvolume\{\textit{entry-label}\}} \hspace{1em} \text{(defined by \texttt{GlsXtrProvideBibTeXFields})} \hspace{1em} \S11.5.2; 536

Accesses the \texttt{volume} field.

\texttt{\textbackslash glsxtrbookindexatendgroup\{\textit{entry-label}\}} \hspace{1em} \text{glossary-bookindex v1.21+} \hspace{1em} \S8.7.1; 441

Used by the bookindex style at the end of a letter group (where the last top-level entry is given by \textit{entry-label}).

\texttt{\textbackslash glsxtrbookindexbetween\{\textit{entry1-label}\}\{\textit{entry2-label}\}} \hspace{1em} \text{glossary-bookindex v1.21+} \hspace{1em} \S8.7.1; 441

Used by the bookindex style between two entries where \texttt{\{entry1-label\}} is the last top-level entry and \texttt{\{entry2-label\}} is the next entry, which is a top-level entry.

\texttt{\textbackslash glsxtrbookindexbookmark\{\textit{group-title}\}\{\textit{bookmark-name}\}} \hspace{1em} \text{glossary-bookindex v1.21+} \hspace{1em} \S8.7.1; 442

Adds a bookmark with \texttt{\textbackslash pdfbookmark}, if supported.

\texttt{\textbackslash glsxtrbookindexcols} \hspace{1em} \textit{initial: 2} \hspace{1em} \text{glossary-bookindex v1.21+} \hspace{1em} \S8.7.1; 438

Expands to the number of columns for the bookindex style.

\texttt{\textbackslash glsxtrbookindexcolspread} \hspace{1em} \text{glossary-bookindex v1.12+} \hspace{1em} \S8.7.1; 439

If not empty this should expand to the option argument for \texttt{multicols}.

\texttt{\textbackslash glsxtrbookindexfirstmark\{\textit{entry-label}\}} \hspace{1em} \text{glossary-bookindex v1.21+} \hspace{1em} \S8.7.1; 443

Used by the bookindex style to obtain the first mark and, if found, format it with \texttt{\textbackslash glsxtrbookindexfirstmarkfmt}.


Command Summary

\makelabel[bookindex]{entry-label}{glossary-bookindex v1.21+}§8.7.1; 444

Used by the bookindex style to format the first mark.

\makelabel[bookindex]{group-title}{glossary-bookindex v1.21+}§8.7.1; 443

Used by the bookindex style to format a group header.

\makelabel[bookindex]{previous level}{level}{parent-label}{group-label}{title}{glossary-bookindex v1.49+}

Formats the sub-group header.

\makelabel[bookindex]{entry-label}{glossary-bookindex v1.21+}§8.7.1; 443

Used by the bookindex style to obtain the last mark and, if found, format it with \glsxtrbookindexlastmarkfmt.

\glsxtrbookindexlastmarkfmt{entry-label}{glossary-bookindex v1.21+}§8.7.1; 444

Used by the bookindex style to format the last mark.

\makelabel[bookindex]{entry-label}{location list}{glossary-bookindex v1.39+}§8.7.1; 440

Used by the bookindex style to display top-level location lists.

\makelabel[bookindex]{entry-label}{glossary-bookindex v1.21+}§8.7.1; 443

Used by the bookindex style to mark an entry in the aux file.

789
Expands to the name of the multicols environment to use.

Used by the bookindex style to display a top-level entry’s name.

Used by the bookindex style to separate a top-level parent and child entry.

Used by the bookindex style to separate a sub-level parent and child entry.

Used by the bookindex style insert \(\langle\text{skip}\rangle\) after a group header.

Used by the bookindex style to display a separator before top-level location lists.

Used by the bookindex style at the end of a letter group (where the last level 1 entry is given by \(\langle\text{entry-label}\rangle\)).
Command Summary

As `\glsxtrbookindexbetween` but for level 1 entries.

```
\glsxtrbookindexsublocation{⟨entry-label⟩}{⟨location list⟩}
glossary-bookindex v1.39+
```

§8.7.1; 440

Used by the `bookindex` style to display child location lists.

```
\glsxtrbookindexsubname{⟨entry-label⟩}
glossary-bookindex v1.21+
```

§8.7.1; 440

Used by the `bookindex` style to display a child entry’s name.

```
\glsxtrbookindexsubprelocation{⟨entry-label⟩}
glossary-bookindex v1.21+
```

§8.7.1; 440

Used by the `bookindex` style to display a separator before child location lists.

```
\glsxtrbookindexsubsubatendgroup{⟨entry-label⟩}
glossary-bookindex v1.21+
```

§8.7.1; 441

Used by the `bookindex` style at the end of a letter group (where the last level 2 entry is given by ⟨entry-label⟩).

```
\glsxtrbookindexsubsubbetween{⟨entry1-label⟩}{⟨entry2-label⟩}
glossary-bookindex v1.21+
```

§8.7.1; 441

As `\glsxtrbookindexbetween` but for level 2 entries.

```
\glsxtrcat initial: general
```

§13; 580

Expands to the default category set by commands like `\glsxtr`.

```
\glsxtr(category)(field)accsupp
```

(user defined)
Expands to the accessibility support command for the given internal field label and category, which is used by `\glsfieldaccsupp`.

\glsextrchecknohyperfirst\{entry-label\} \hspace{1cm} glossaries-extra v1.07+

Sets hyper=false if the nohyperfirst attribute is set.

\glsextrclearlabelprefixes \hspace{1cm} glossaries-extra-bib2gls v1.37+

Clears the list of known prefixes.

\GlsXtrClearUnsetBuffer \hspace{1cm} glossaries-extra v1.49+

Locally clears the buffer, but doesn’t stop buffering.

\glsextrcombiningdiacriticIIIrules \hspace{1cm} glossaries-extra-bib2gls v1.27+

Expands to the third set of combining diacritic sort rules.

\glsextrcombiningdiacriticIIrules \hspace{1cm} glossaries-extra-bib2gls v1.27+

Expands to the second set of combining diacritic sort rules.

\glsextrcombiningdiacriticIrules \hspace{1cm} glossaries-extra-bib2gls v1.27+

Expands to the first set of combining diacritic sort rules.

\glsextrcombiningdiacriticIVrules \hspace{1cm} glossaries-extra-bib2gls v1.27+

Expands to the fourth set of combining diacritic sort rules.
\textit{Command Summary}

\begin{itemize}
  \item \texttt{\glsxtrcombiningdiacriticrules} \quad \texttt{glossaries-extra-bib2gls v1.27+} \hfill 542
    Expands to all the combining diacritic sort rules.
  \item \texttt{\glsxtrcontrolrules} \quad \texttt{glossaries-extra-bib2gls v1.27+} \hfill 542
    Expands to control character sort rules.
  \item \texttt{\glsxtrcopytglossary\{entry-label\}\{glossary-type\}} \quad \texttt{ modifier: *} \hfill §8; 376
    Copies the entry to the internal glossary list for the given glossary. The starred version performs a global change. The unstarred version can be localised. Only for use with the “unsrt” family of commands.
  \item \texttt{\glsxtr\{counter\}lofmt\{location\}\{title\}} \quad \texttt{(user defined)} \hfill §11.5.6; 558
    Used by \texttt{\glsxtrdisplayloctime} for format a location where the counter matches \texttt{\langle counter\rangle}.
  \item \texttt{\glsxtrcurrencyrules} \quad \texttt{glossaries-extra-bib2gls v1.27+} \hfill 545
    Expands to currency character sort rules.
  \item \texttt{\glsxtrcurrentfield} \quad \texttt{glossaries-extra v1.49+} \hfill §5.5.4; 252
    Placeholder command for use in post-link hooks. This expands to empty if the calling command was one of the \texttt{\gls}-like commands or it was one of the inline full form commands, otherwise it will expand to the name of the key associated with the \textit{singular} form of the command.
  \item \texttt{\glsxtrcurrentmglscsname} \quad \texttt{glossaries-extra v1.48+} \hfill §7.5; 348
\end{itemize}
Placeholder command for use in multi-entry hooks, this expands to the control sequence name of the calling command.

\glsxtrdefaultentrytextfmt{\textit{text}} \text{glossaries-extra v1.49+} §5.5.3; 243

Default inner formatting. Initialised to just do \textit{text}.

\GlsXtrDefaultResourceOptions \text{glossaries-extra v1.40+} §11; 513

Expands to default resource options.

\glsxtrdefaultrevert{\textit{text}} \text{glossaries-extra v1.49+} 133

The default definition of \glsxtrrevert. Simply does \textit{text}.

\GLSxtrdefaultsubsequentfmt{\textit{entry-label}}{\textit{insert}} \text{glossaries-extra v1.49+} 174

The default all caps subsequent format style that only shows the short form and insert (with support for \textit{innertextformat}).

\GLSxtrdefaultsubsequentfmt{\textit{entry-label}}{\textit{insert}} \text{glossaries-extra v1.17+} 174

The default sentence case subsequent format style that only shows the short form and insert (with support for \textit{innertextformat}).

\glsxtrdefaultsubsequentfmt{\textit{entry-label}}{\textit{insert}} \text{glossaries-extra v1.17+} 174

The default subsequent format style that only shows the short form and insert (with support for \textit{innertextformat}).

\GLSxtrdefaultsubsequentplfmt{\textit{entry-label}}{\textit{insert}} \text{glossaries-extra v1.49+} 174

794
The default all caps subsequent plural format style that only shows the short form and insert (with support for \textit{innertextformat}).

\begin{verbatim}
\Glsxtrdefaultsubsequentplfmt\{⟨entry-label⟩\}\{⟨insert⟩\}   glossaries-extra v1.17+
\end{verbatim}

The default sentence case subsequent plural format style that only shows the short form and insert (with support for \textit{innertextformat}).

\begin{verbatim}
\glsxtrdefaultsubsequentplfmt\{⟨entry-label⟩\}\{⟨insert⟩\}   glossaries-extra v1.17+
\end{verbatim}

The default subsequent plural format style that only shows the short form and insert (with support for \textit{innertextformat}).

\begin{verbatim}
\glsxtrdeffield\{⟨entry-label⟩\}\{⟨field-label⟩\}\{⟨value⟩\}   glossaries-extra v1.12+
\end{verbatim}

Like \texttt{\GlsXtrSetField} but doesn’t perform any existence checks.

\begin{verbatim}
\GlsXtrDefineAbbreviationShortcuts
\end{verbatim}

Used by \texttt{shortcuts=abbreviations} and \texttt{shortcuts=all}. This command redefines itself to do nothing because it can only be used once.

\begin{verbatim}
\GlsXtrDefineAcronymShortcuts   glossaries-extra v1.17+
\end{verbatim}

Used by \texttt{shortcuts=ac} and \texttt{shortcuts=acother}. This command redefines itself to do nothing because it can only be used once.

\begin{verbatim}
\GlsXtrDefineOtherShortcuts
\end{verbatim}

Used by \texttt{shortcuts=other} and \texttt{shortcuts=all}. This command redefines itself to do nothing because it can only be used once.
\texttt{\glsxtrdetoklocation}{\langle location \rangle} \quad \texttt{glossaries-extra v1.21+} \quad \S 11.4; 522

Just expands to \langle location \rangle by default but may be redefined to help protect awkward characters.

\texttt{\glsxtdigitrules} \quad \texttt{glossaries-extra-bib2gls v1.27+} \quad 545

Expands to 0–9 digit character sort rules (includes superscript and subscript digits).

\texttt{\glsxtrdiscardperiod}{\langle entry-label \rangle}{\langle discarded \rangle}{\langle no discard \rangle}{\langle token \rangle} \quad \texttt{glossaries-extra} \quad \S 5.5.4; 247

If \langle token \rangle is a full stop and the entry’s category attributes indicate that a full stop should be discarded (such as \texttt{discardperiod}), then \langle discarded \rangle is performed, otherwise \langle no discard \rangle is done and the \langle token \rangle is processed. The actual test to determine if \langle token \rangle is a full stop is performed by \texttt{\glsxtrimperiod}. This command is used in post-link hooks.

\texttt{\glsxtrdiscardperiodretainfirstuse}{\langle entry-label \rangle}{\langle discarded \rangle}{\langle no discard \rangle}{\langle token \rangle} \quad \texttt{glossaries-extra v1.49+} \quad \S 5.5.4; 247

Used to discard a following full stop when the \texttt{retainfirstuseperiod} attribute is set.

\texttt{\GlsXtrDiscardUnsetBuffering} \quad \texttt{glossaries-extra v1.42+} \quad \S 5.10.1; 285

Discards the pending buffer and restores \texttt{\glsunset}.

\texttt{\glsxtrdisplayendloc}{\langle format \rangle}{\langle location \rangle} \quad \texttt{glossaries-extra v1.12+} \quad \S 8.6.3; 424

Used to display an end location from an explicit range.

\texttt{\glsxtrdisplayendlochook} \quad \texttt{glossaries-extra v1.12+} \quad \S 8.6.3; 425

796
Hook used by \glsxtrdisplayendloc.

\glsxtrdisplaylocnameref{(prefix){\langle counter\rangle}{\langle format\rangle}{\langle location\rangle}{\langle title\rangle}}{(href){\langle hcounter\rangle}{\langle file\rangle}}
glossaries-extra-bib2gls v1.37+

§11.5.6; 556

Used to display records created with record=nameref.

\glsxtrdisplaysingleloc{\langle format\rangle}{\langle location\rangle}
glossaries-extra v1.12+

§8.6.3; 424

Used to display a single location.

\glsxtrdisplaystartloc{\langle format\rangle}{\langle location\rangle}
glossaries-extra v1.12+

§8.6.3; 424

Used to display a start location from an explicit range.

\glsxtrdisplaysupploc{(prefix){\langle counter\rangle}{\langle format\rangle}{\langle src\rangle}{\langle location\rangle}}
 modifier: * glossaries-extra-bib2gls v1.36+

§11.5.5; 555

Like \glsnoidxdisplayloc but used for supplementary locations.

\glsxtrdoautoindexname{(entry-label)}{(attribute)}

§12; 574

Used to automatically index (using \glsxtrautoindex) the entry’s name, if the given attribute is set for the entry’s category.

\glsxtrdopostpunc{\langle code\rangle}{\langle token\rangle}
glossaries-extra v1.49+

§5.5.4; 248

If \langle token\rangle is a recognised punctuation character (see \glsxtrifnextpunc) this does the punctuation character and then \langle code\rangle, otherwise if does \langle code\rangle followed by \langle token\rangle.

\glsxtrdowrglossaryhook{(entry-label)}
glossaries-extra v1.49+

§5.8; 263
Hook used whenever an entry is indexed. Does nothing by default.

\GlsXtrDualBackLink{⟨text⟩}{⟨entry-label⟩} glossaries-extra-bib2gls v1.30+

§11.5.7; 561

Adds a hyperlink to the given entry’s dual (whose label is stored in the field given by \GlsXtrDualField) with the given hyperlink text.

\GlsXtrDualField

§11.5.7; 562

Expands to the internal field label used by \GlsXtrDualBackLink.

\glsxtrdeffield{⟨entry-label⟩}{⟨field-label⟩}{⟨value⟩} glossaries-extra v1.12+

§3.5; 38

Like \glsxtrdeffield but (protected) expands ⟨value⟩.

\glsxtremrevert{⟨text⟩} glossaries-extra v1.49+

158

The definition of \glsxtrrevert used by the emphasized (“em”) abbreviation styles. Uses \textup.

\glsxtremsuffix

158

The plural suffix used by the emphasized (“em”) abbreviation styles.

\GlsXtrEnableEntryCounting{⟨category-list⟩}{⟨trigger-value⟩}

§6.1; 308

Enables entry counting for the given list of categories with the given trigger value (which must be an integer).

\GlsXtrEnableEntryUnitCounting{⟨category-list⟩}{⟨trigger-value⟩}{⟨counter⟩}

§6.1; 313

798
Command Summary

Enables unit entry counting for the given list of categories with the given trigger value (which must be an integer) and the associated counter.

\GlsXtrEnableIndexFormatOverride \vspace{10pt} \hspace{30pt} (preamble only) \vspace{10pt} \hspace{70pt} \S12; 576

Allows the format key to override the attribute value.

\GlsXtrEnableInitialTagging\{\langle categories\rangle\}\{\langle cs\rangle\} \hspace{30pt} \textit{modifier: *} \vspace{10pt} \hspace{70pt} \S4.4; 56

Robustly defines the command \langle cs\rangle to accept a single argument, which is a letter (or letters) that needs to be tagged. The unstarred version triggers an error if \langle cs\rangle is already defined. The unstarred version will redefine \langle cs\rangle if it already exists.

\GlsXtrEnableLinkCounting\[\langle parent counter\rangle]\{\langle categories\rangle\} \hspace{10pt} \textit{glossaries-extra v1.26+} \vspace{10pt} \hspace{70pt} \S6.2; 317

Enables link counting for the given categories.

\GlsXtrEnableOnTheFly \vspace{10pt} \hspace{30pt} \textit{modifier: *} \vspace{10pt} \hspace{70pt} \S13; 578

Enables on the fly commands, such as \glsxtr.

\GlsXtrEnablePreLocationTag\{\langle page tag\rangle\}\{\langle pages tag\rangle\} \hspace{10pt} \textit{glossaries-extra v1.04+} \vspace{10pt} \hspace{70pt} \S8.6.3; 423

Enables the location list tag.

\glsxtrenablerecordcount \vspace{10pt} \hspace{30pt} \textit{glossaries-extra v1.21+} \vspace{10pt} \hspace{70pt} \S11.4; 527

Redefines the \gls-like commands (except \glsdisp) to use the analogous record count commands (\rgls etc).
Command Summary

\glsxtrendfor glosaries-extra v1.24+ §5.13; 301

When used within \glsxtrforcsvfield signifies that the loop should break at the end of the current iteration.

\Glsxtrentryfmt{⟨entry-label⟩}{⟨text⟩} glosaries-extra v1.49+ §5.12.2; 298

As \glsxtrentryfmt but converts ⟨text⟩ to sentence case.

\glsxtrentryfmt{⟨entry-label⟩}{⟨text⟩} glosaries-extra v1.12+ §5.12.2; 296

Does \⟨csname⟩{⟨text⟩} where the control sequence name ⟨csname⟩ is obtained from the field given by \GlsXtrFmtField. If hyperref has been loaded and this command will expand to \glsxtrpdfentryfmt{⟨entry-label⟩}{⟨text⟩} in a PDF bookmark.

\glsxtrentryparentname{⟨entry-name⟩} glosaries-extra v1.39+ §5.11; 291

Expands to the name field of the given entry’s parent or does nothing if the entry doesn’t have the parent field set or isn’t defined.

\glsxtrequationlocfmt{⟨location⟩}{⟨title⟩} glossaries-extra-bib2gls v1.42+ §11.5.6; 558

Used by \glsxtrdisplaylocnameref to format a location where the counter is equation.

\GlsXtrExpandedFmt{⟨cs⟩}{⟨content⟩} glossaries-extra v1.30+ §5.5; 236

Fully-expands ⟨content⟩ and passes it to ⟨cs⟩, which must be a command that takes a single argument.

\glsxtrfielddolistloop{⟨entry-label⟩}{⟨field⟩} glosaries-extra v1.12+ §5.14; 304

Iterates over the given field’s value using etoolbox’s \dolistcsloop.
Iterates over the given field’s value using etoolbox’s \forlistcsloop.

\glsxtrfieldforlistloop\{⟨entry-label⟩\}{⟨field⟩}\{⟨handler-cs⟩\}
glossaries-extra v1.12+

§5.14; 304

Formats the comma-separated list stored in the given field (identified by its internal label) for the entry identified by ⟨entry-label⟩ using datatool-base’s \DTLformatlist. This command uses \glsxtrifhasfield so the complete list can be obtained with \glscurrentfield-value. This adds implicit grouping. There is no starred version.

\glsxtrfieldformatcsvlist\{⟨entry-label⟩\}{⟨field-label⟩}
glossaries-extra v1.42+

§5.13; 302

Formats the value of the given field, which should be an etoolbox internal list, using the same list handler macro as datatool’s \DTLformatlist.

\glsxtrfieldformatlist\{⟨entry-label⟩\}{⟨field-label⟩}
glossaries-extra v1.42+

§5.14; 304

Uses etoolbox’s \ifinlistcs to determine if ⟨item⟩ is in the list stored in the given field.

\glsxtrfieldifinlist\{⟨entry-label⟩\}{⟨field⟩}\{⟨item⟩\}{⟨true⟩}\{⟨false⟩\}
glossaries-extra v1.12+

§5.14; 304

Appends ⟨value⟩ to the given field using etoolbox’s \listcsadd.

\glsxtrfieldlistadd\{⟨entry-label⟩\}{⟨field⟩}\{⟨value⟩\}
glossaries-extra v1.12+

§3.5; 39

\glsxtrfieldlsteadd\{⟨entry-label⟩\}{⟨field⟩}\{⟨value⟩\}
glossaries-extra v1.12+

§3.5; 39

Appends ⟨value⟩ to the given field using etoolbox’s \listcseadd.

\glsxtrfieldlistgadd\{⟨entry-label⟩\}{⟨field⟩}\{⟨value⟩\}
glossaries-extra v1.12+

§3.5; 39
Command Summary

Appends \textit{value} to the given field using etoolbox’s \texttt{\textbackslash listcsgadd}.

\begin{verbatim}
\texttt{\textbackslash glsxtrfieldlistxadd\{\texttt{entry-label}\}\{\texttt{field}\}\{\texttt{value}\}}
glossaries-extra v1.12+
\end{verbatim}

§3.5; 39

Appends \textit{value} to the given field using etoolbox’s \texttt{\textbackslash listsxadd}.

\begin{verbatim}
\texttt{\textbackslash glsxtrfieldtitlecase\{\texttt{entry-label}\}\{\texttt{field-label}\}}
glossaries-extra v1.12+
\end{verbatim}

§5.11; 291

As \texttt{\textbackslash glsxtrusefield} but converts the field value to title case.

\begin{verbatim}
\texttt{\textbackslash glsxtrfieldtitlecasesc\{\texttt{content}\}}
glossaries-extra v1.07+
\end{verbatim}

§5.11; 291

Converts \textit{content} to title case (expanding the first token once). Uses \texttt{\textbackslash glscapitalise} words, if defined, otherwise uses \texttt{\textbackslash capitalisewords}.

\begin{verbatim}
\texttt{\textbackslash glsxtrfieldxifinlist\{\texttt{entry-label}\}\{\texttt{field}\}\{\texttt{item}\}\{\texttt{true}\}\{\texttt{false}\}}
glossaries-extra v1.12+
\end{verbatim}

§5.14; 305

Uses etoolbox’s \texttt{\textbackslash xifinlistcs} to determine if \textit{item} is in the list stored in the given field.

\begin{verbatim}
\texttt{\textbackslash glsxtrfirstscfont\{\texttt{text}\}}
glossaries-extra v1.04+
\end{verbatim}

§5.12.2; 298

Maintained for backwards-compatibility used to typeset \textit{text} in small capitals (\texttt{\textbackslash textsc}) for the “sc” abbreviation styles on first use.

\begin{verbatim}
\texttt{\textbackslash glsxtrfirstsmfont\{\texttt{text}\}}
glossaries-extra v1.04+
\end{verbatim}

Maintained for backwards-compatibility used to typeset \textit{text} in a smaller font (\texttt{\textbackslash textsmaller}) for the “sm” abbreviation styles on first use.

\begin{verbatim}
\texttt{\textbackslash Gl sxtrfmt\{\texttt{options}\}\{\texttt{entry-label}\}\{\texttt{text}\}}
glossaries-extra v1.49+
\end{verbatim}

§5.12.2; 298
As \glsxtrfmt but applies a sentence case change to \textit{⟨text⟩}.

\glsxtrfmt\[⟨options⟩]\{⟨entry-label⟩\}{⟨text⟩}  
\textit{glossaries-extra v1.12+}  
\textit{§5.12.2; 295}

Behaves like \glslink\[⟨options⟩]\{⟨entry-label⟩\}{⟨csname⟩}\{⟨text⟩\}{⟨insert⟩} where the control sequence name \textit{⟨csname⟩} is obtained from the field given by \GlsXtrFmtField. The actual format of the link text is governed by \glsxtrfmtdisplay.

\GlsXtrFmt*[⟨options⟩]\{⟨entry-label⟩\}{⟨text⟩}\{⟨insert⟩\}  
\textit{glossaries-extra v1.49+}  
\textit{§5.12.2; 298}

As \glsxtrfmt* but applies a sentence case change to \textit{⟨text⟩}.

\glsxtrfmt*[⟨options⟩]\{⟨entry-label⟩\}{⟨text⟩}\{⟨insert⟩\}  
\textit{glossaries-extra v1.23+}  

As the unstarred version \glsxtrfmt but accepts the final \textit{⟨insert⟩} option.

\GlsXtrFmtDefaultOptions  
\textit{initial: noindex}  
\textit{glossaries-extra v1.12+}  
\textit{§5.1.1; 188}

Expands to the default options for \glsxtrfmt.

\glsxtrfmtdisplay\{⟨csname⟩\}{⟨text⟩}\{⟨insert⟩\}  
\textit{glossaries-extra v1.23+}  
\textit{§5.12.2; 296}

Formats the link text used in \glsxtrfmt.

\glsxtrfmtexternalnameref\{⟨target⟩\}{⟨format⟩}\{⟨title⟩\}\{⟨file⟩\}  
\textit{glossaries-extra–bib2gls v1.37+}  
\textit{§11.5.6; 559}

Used by \glsxtrnamereflink to create an external location hyperlink.

\GlsXtrFmtField  
\textit{initial: useri}  
\textit{glossaries-extra v1.12+}  
\textit{§5.12.2; 295}
Command Summary

Expands to the name of the used by \glsxtrfmt.

\[\text{\glsxtrfmtinternalnameref\{\langle target\rangle\}\{\langle format\rangle\}\{\langle file\rangle\}}\]
glossaries-extra-bib2gls v1.37+

§11.5.6; 559

Used by \glsxtrnamereflink to create an internal location hyperlink.

\[\text{\glsxtrfootnotedescname}\]
glossaries-extra v1.42+

144

Expands to the name value for styles like short\text{-footnote\text{-desc}}.

\[\text{\glsxtrfootnotedescsort}\]
glossaries-extra v1.42+

145

Expands to the sort value for styles like short\text{-footnote\text{-desc}}.

\[\text{\glsxtrfootnotelongformat\{\langle entry-label\rangle\}\{\langle fmt-cs\rangle\}}\]
glossaries-extra v1.49+

145

Used in the footnote text to format the singular long form.

\[\text{\glsxtrfootnotelongplformat\{\langle entry-label\rangle\}\{\langle fmt-cs\rangle\}}\]
glossaries-extra v1.49+

146

Used in the footnote text to format the plural long form.

\[\text{\glsxtrfootnotename}\]
glossaries-extra v1.25+

144

Expands to the name value for styles like short\text{-footnote}.

\[\text{\glsxtrforcsvfield\{\langle entry-label\rangle\}\{\langle field-label\rangle\}\{\langle handler cs\rangle\}}\]
modifier: *
glossaries-extra v1.24+

§5.13; 301
Command Summary

Iterates over the comma-separated list stored in the given field (identified by its internal label) for the entry identified by \langle entry-label \rangle and performs \langle handler cs \rangle \{ \langle element \rangle \} for each element of the list. This command uses \glsxtrifhasfield so the complete list can be obtained with \glscurrentfieldvalue. The unstarred version adds implicit grouping. The starred version doesn’t.

\GlsXtrForeignText{\langle entry-label \rangle}{\langle text \rangle} \quad \text{glossaries-extra v1.32+} \quad \S5.12.1; 293

If the entry given by \langle entry-label \rangle has the field identified by \GlsXtrForeignTextField then \langle text \rangle will be encapsulated according to the language tag stored in that field (using tracklang’s interface).

\GlsXtrForeignTextField \quad \text{glossaries-extra v1.32+} \quad \S5.12.1; 293

Expands to the internal field label used by \GlsXtrForeignText.

\GlsXtrFormatLocationList{\langle location list \rangle} \quad \S8.6.3; 423

Used by \glossaryentrynumbers to encapsulate the entire location list in the glossary.

\GlsXtrForUnsetBufferedList{\langle handler-cs \rangle} \quad \text{glossaries-extra v1.31+} \quad \S5.10.1; 285

Iterates over the labels stored in the current buffer.

\glsxtrfractionrules \quad \text{glossaries-extra-bib2gls v1.27+} \quad 546

Expands to the number forms fraction character sort rules.

\GLSxtrfull[\langle options \rangle]{\langle entry-label \rangle}[\langle insert \rangle] \quad \text{modifiers: * + \langle alt-mod \rangle} \quad \S4.3; 54

As \glsxtrfull but converts the link text to all caps.

805
Command Summary

\Glsxtrfull{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩  

§4.3; 53

As \glsxtrfull but converts the first character of the link text to uppercase (for the start of a sentence) using \makefirstuc.

\Glsxtrfull{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩  

§4.3; 53

References the entry identified by ⟨entry-label⟩. The text produced is obtained from the short and long values, formatted according to the abbreviation style associated with the entry’s category. The ⟨insert⟩ argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. The format produced by this command may not match the format produced by the first use of \gls{⟨entry-label⟩}, depending on the abbreviation style. For the first optional argument, see \glslink options.

\GLSxtrfullformat{⟨entry-label⟩}{⟨insert⟩} glossaries-extra v1.49+

173

The all caps singular display full form (defined by the abbreviation style).

\GLSxtrfullformat{⟨entry-label⟩}{⟨insert⟩} glossaries-extra v1.49+

172

The sentence case singular display full form (defined by the abbreviation style).

\glsxtrfullformat{⟨entry-label⟩}{⟨insert⟩} glossaries-extra v1.49+

172

The singular display full form (defined by the abbreviation style).

\GLSxtrfullpl{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩  

§4.3; 54

As \glsxtrfullpl but converts the link text to all caps.

\Glsxtrfullpl{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩  

§4.3; 54
As \glsxtrfullpl but converts the first character of the link text to uppercase (for the start of a sentence) using  \makefirstuc.

\glsxtrfullpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} modifiers: * + ⟨alt-mod⟩

References the entry identified by ⟨entry-label⟩. The text produced is obtained from the shortplural and longplural values, formatted according to the abbreviation style associated with the entry’s category. The ⟨insert⟩ argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. The format produced by this command may not match the format produced by the first use of \glspl{⟨entry-label⟩}, depending on the abbreviation style. For the first optional argument, see \glslink options.

\GLSxtrfullplformat{⟨entry-label⟩}{⟨insert⟩} glossaries-extra v1.49+

The all caps plural display full form (defined by the abbreviation style).

\Glsxtrfullplformat{⟨entry-label⟩}{⟨insert⟩} glossaries-extra v1.49+

The sentence case plural display full form (defined by the abbreviation style).

\glsxtrfullplformat{⟨entry-label⟩}{⟨insert⟩} glossaries-extra v1.49+

The plural display full form (defined by the abbreviation style).

\glsxtrfullsaveinsert{⟨entry-label⟩}{⟨insert⟩} glossaries-extra v1.49+

Implemented at the start of all the inline full form commands like \glsxtrfull to save the \glsinsert placeholder. By default, this just does \glsxtrsaveinsert.

\glsxtrfullsep{⟨entry-label⟩}

Separator used by the parenthetical inline full form and also for some display full forms.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrgenabbrvfmt</td>
<td>The display format used by \glsentryfmt for entries that have the <code>short</code> field set and have the <code>regular</code> attribute set to <code>false</code>.</td>
</tr>
<tr>
<td>\glsxtrgenentrytextfmt</td>
<td>Redefined by the \gls-like and \glstext-like hooks to set up the inner formatting. Initialised to \glsxtrdefaultentrytextfmt.</td>
</tr>
<tr>
<td>\glsxtrGeneralInitRules</td>
<td>A shortcut that expands to the ignorable rules, combining diacritic rules, hyphen rules, general punctuation rules, digit rules, and fraction rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinAtoGrules</td>
<td>Expands to the A–G subset of General Latin I sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinAtoMrules</td>
<td>Expands to the A–M subset of General Latin I sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinHtoMrules</td>
<td>Expands to the H–M subset of General Latin I sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinIIIRules</td>
<td>Expands to the third set of General Latin sort rules.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinIIrules</td>
<td>Expands to the second set of General Latin sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinIrules</td>
<td>Expands to the first set of General Latin sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinIVrules</td>
<td>Expands to the fourth set of General Latin sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinNtoSrules</td>
<td>Expands to the N–S subset of General Latin I sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinNtoZrules</td>
<td>Expands to the N–Z subset of General Latin I sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinTtoZrules</td>
<td>Expands to the T–Z subset of General Latin I sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinVIIIrules</td>
<td>Expands to the eighth set of General Latin sort rules.</td>
</tr>
<tr>
<td>\glsxtrGeneralLatinVIIrules</td>
<td>Expands to the seventh set of General Latin sort rules.</td>
</tr>
</tbody>
</table>
Command Summary

\texttt{\textbackslash glsxtrGeneralLatinVIrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+} \hspace{1cm} 547

Expands to the sixth set of General Latin sort rules.

\texttt{\textbackslash glsxtrGeneralLatinVrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+} \hspace{1cm} 547

Expands to the fifth set of General Latin sort rules.

\texttt{\textbackslash glsxtrgeneralpuncaccentsrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.49+} \hspace{1cm} 544

Punctuation accent subset of \texttt{\textbackslash glsxtrgeneralpuncIrules}.

\texttt{\textbackslash glsxtrgeneralpuncbracketrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.49+} \hspace{1cm} 544

Punctuation bracket subset of \texttt{\textbackslash glsxtrgeneralpuncIrules}.

\texttt{\textbackslash glsxtrgeneralpuncIIrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+} \hspace{1cm} 545

Expands to the second set of general punctuation (including currency) sort rules.

\texttt{\textbackslash glsxtrgeneralpuncIrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+} \hspace{1cm} 544

Expands to the first set of general punctuation (including currency) sort rules.

\texttt{\textbackslash glsxtrgeneralpuncmarksrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.49+} \hspace{1cm} 544

Punctuation mark subset of \texttt{\textbackslash glsxtrgeneralpuncIrules}.

\texttt{\textbackslash glsxtrgeneralpuncquoterules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.49+} \hspace{1cm} 544

Punctuation quote subset of \texttt{\textbackslash glsxtrgeneralpuncIrules}.
Command Summary

\texttt{\textbackslash glsxtrgeneralpuncrules} \hspace{1cm} \texttt{glossaries\textasciitilde extra\textasciitilde bib2gls v1.27+}

Expands to all sets of general punctuation sort rules.

\texttt{\textbackslash glsxtrgeneralpuncsignrules} \hspace{1cm} \texttt{glossaries\textasciitilde extra\textasciitilde bib2gls v1.49+}

Punctuation sign subset of \texttt{\textbackslash glsxtrgeneralpuncIrules}.

\texttt{\textbackslash glsxtrgetgrouptitle\{\langle group-label\rangle\}\{\langle cs\rangle\}} \hspace{1cm} \texttt{glossaries\textasciitilde extra v1.14+}

\S 8.6.4; 426

Obtains the title corresponding to the group identified by \texttt{\langle group-label\rangle} and stores the result in the control sequence \texttt{\langle cs\rangle}.

\texttt{\textbackslash glsxtrgroupfield} \hspace{1cm} \texttt{\textit{initial}: group} \hspace{1cm} \texttt{glossaries\textasciitilde extra v1.21+}

\S 8.4.1; 389

Expands to the internal field label used to store the group label (requires \texttt{record}).

\texttt{\textbackslash glsxtrglossentry\{\langle entry-label\rangle\}} \hspace{1cm} \texttt{glossaries\textasciitilde extra v1.21+}

\S 8.5; 414

Used for standalone entries to display the name with \texttt{\textbackslash glossentryname}, with appropriate hooks.

\texttt{\textbackslash glsxtrglossentryother\{\langle header\rangle\}\{\langle entry-label\rangle\}\{\langle field-label\rangle\}} \hspace{1cm} \texttt{glossaries\textasciitilde extra v1.22+}

\S 8.5; 415

Like \texttt{\textbackslash glsxtrglossentry} but uses the given field instead of \texttt{name}.

\texttt{\textbackslash GLSxtrheadfirst\{\langle entry-label\rangle\}} \hspace{1cm} \texttt{glossaries\textasciitilde extra v1.49+}

\S 5.3.3; 226

Used to display the all caps entry’s \texttt{first} field in the page header.
\Glsxtrheadfirst{(entry-label)}

Used to display the sentence case entry’s first field in the page header (converts to all caps if headuc attribute is true).

\glsxtrheadfirst{(entry-label)}

Used to display the entry’s first field in the page header (converts to all caps if headuc attribute is true).

\GLSxtrheadfirstplural{(entry-label)}

glossaries-extra v1.49+

Used to display the all caps entry’s firstplural field in the page header.

\Glsxtrheadfirstplural{(entry-label)}

Used to display the sentence case entry’s firstplural field in the page header (converts to all caps if headuc attribute is true).

\glsxtrheadfirstplural{(entry-label)}

Used to display the entry’s firstplural field in the page header (converts to all caps if headuc attribute is true).

\GLSxtrheadfull{(entry-label)}

glossaries-extra v1.49+

Used to display the entry’s all caps full form in the page header.

\Glsxtrheadfull{(entry-label)}

glossaries-extra v1.02+

Used to display the entry’s sentence case full form in the page header (converts to all caps if headuc attribute is true).
Command Summary

\glsxtrheadfull{⟨entry-label⟩}
glossaries-extra v1.02+ $\S5.3.3; 221$

Used to display the entry’s full form in the page header (converts to all caps if headuc attribute is true).

\GLSxtrheadfullpl{⟨entry-label⟩}
glossaries-extra v1.49+ $\S5.3.3; 223$

Used to display the entry’s all caps full plural form in the page header.

\Glsxtrheadfullpl{⟨entry-label⟩}
glossaries-extra v1.02+ $\S5.3.3; 222$

Used to display the entry’s sentence case full plural form in the page header (converts to all caps if headuc attribute is true).

\glsxtrheadfullpl{⟨entry-label⟩}
glossaries-extra v1.02+ $\S5.3.3; 222$

Used to display the entry’s full plural form in the page header (converts to all caps if headuc attribute is true).

\GLSxtrheadlong{⟨entry-label⟩}
glossaries-extra v1.49+ $\S5.3.3; 220$

The behaviour of \GLSfmtlong when it occurs in a page header.

\Glsxtrheadlong{⟨entry-label⟩}
glossaries-extra v1.02+ $\S5.3.3; 220$

The behaviour of \Glsfmtlong when it occurs in a page header.

\glsxtrheadlong{⟨entry-label⟩}
glossaries-extra v1.02+ $\S5.3.3; 220$

The behaviour of \glsfmtlong when it occurs in a page header.
The behaviour of \texttt{\GLSfmtlongpl} when it occurs in a page header.

\texttt{\Glsxtrheadlongpl\{⟨entry-label⟩\}} \hfill glossaries-extra v1.02+

The behaviour of \texttt{\Glsfmtlongpl} when it occurs in a page header.

\texttt{\glsxtrheadlongpl\{⟨entry-label⟩\}} \hfill glossaries-extra v1.02+

The behaviour of \texttt{\glsfmtlongpl} when it occurs in a page header.

\texttt{\GLSxtrheadname\{⟨entry-label⟩\}} \hfill glossaries-extra v1.49+

Used to display the all caps entry’s \texttt{name} field in the page header.

\texttt{\Glsxtrheadname\{⟨entry-label⟩\}} \hfill glossaries-extra v1.21+

Used to display the sentence case entry’s name in the page header (converts to all caps if \texttt{headuc} attribute is true).

\texttt{\glsxtrheadname\{⟨entry-label⟩\}} \hfill glossaries-extra v1.21+

Used to display the entry’s name in the page header (converts to all caps if \texttt{headuc} attribute is true).

\texttt{\GLSxtrheadplural\{⟨entry-label⟩\}} \hfill glossaries-extra v1.49+

Used to display the all caps entry’s \texttt{plural} field in the page header.
\glsxtrheadplural{(entry-label)}

Used to display the sentence case entry’s plural field in the page header (converts to all caps if headuc attribute is true).

\glsxtrheadplural{(entry-label)}

Used to display the entry’s plural field in the page header (converts to all caps if headuc attribute is true).

\GLSxtrheadshort{(entry-label)}

The behaviour of \GLSfmtshort when it occurs in a page header.

\GLSxtrheadshort{(entry-label)}

The behaviour of \GLSfmtshort when it occurs in a page header.

\glsxtrheadshort{(entry-label)}

The behaviour of \glsfmtshort when it occurs in a page header.

\GLSxtrheadshortpl{(entry-label)}

The behaviour of \GLSfmtshortpl when it occurs in a page header.

\GLSxtrheadshortpl{(entry-label)}

The behaviour of \GLSfmtshortpl when it occurs in a page header.
\texttt{\textbackslash glsxtrheadshortpl\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.49+ \hfill \texttt{\textbackslash GLSxtrheadtext\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.49+

The behaviour of \texttt{\textbackslash glsfmtshortpl} when it occurs in a page header.

\texttt{\textbackslash GLSxtrheadtext\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.49+

Used to display the all caps entry’s text field in the page header.

\texttt{\textbackslash GLSxtrheadtext\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.49+

Used to display the sentence case entry’s text field in the page header (converts to all caps if headuc attribute is true).

\texttt{\textbackslash glsxtrheadtext\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.49+

Used to display the entry’s text field in the page header (converts to all caps if headuc attribute is true).

\texttt{\textbackslash GLSXTRhiername\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.37+ \hfill \texttt{\textbackslash GLSxtrhiername\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.37+

Displays the entry’s hierarchical name where each name is converted to uppercase.

\texttt{\textbackslash GLSXTRhiername\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.37+ \hfill \texttt{\textbackslash GLSxtrhiername\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.37+

Displays the entry’s hierarchical name where the first name is converted to uppercase.

\texttt{\textbackslash GLSxtrhiername\{\langle entry-label\rangle\}} \hfill glossaries-extra v1.37+

Displays the entry’s hierarchical name where each element name has its first character converted to uppercase.
## Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrhiernamelabel</td>
<td>Displays the entry’s hierarchical name using sentence case.</td>
<td>glossaries-extra v1.37+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtrhiernamelabel</td>
<td>Displays the entry’s hierarchical name.</td>
<td>glossaries-extra v1.37+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtrhiernamesep</td>
<td>Separator used by commands like \glsxtrhiernamelabel.</td>
<td>glossaries-extra v1.37+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtrhyphenrules</td>
<td>Expands to hyphen character sort rules.</td>
<td>glossaries-extra-bib2gls v1.27+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtrhyphenrules</td>
<td>Expands to hyphen character sort rules.</td>
<td>glossaries-extra-bib2gls v1.27+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtrhyphenrulesinitial: \glsxtrabbrvpluralsuffix</td>
<td>The plural suffix used by the &quot;hyphen&quot; abbreviation styles (such as short-hyphen-long-hyphen).</td>
<td>glossaries-extra v1.17+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtridentifyglslike</td>
<td>Used to inform bib2gls to include the given command when it searches for dependencies.</td>
<td>glossaries-extra v1.37+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtrifallcaps</td>
<td>Shortcut for \glscapscase{not all caps}{not all caps}{all caps}.</td>
<td>glossaries-extra v1.49+</td>
<td>5.11; 292</td>
</tr>
<tr>
<td>\glsxtrifcounttrigger</td>
<td>Shortcut for \glscapscase{not all caps}{not all caps}{all caps}.</td>
<td>glossaries-extra v1.49+</td>
<td>5.11; 292</td>
</tr>
</tbody>
</table>
Does \textit{true} if the entry’s total use count at the end of the previous run exceeds the trigger value assigned to the entry’s category, otherwise does \textit{false}.

\glxtrifcustomdiscardperiod\{\textit{true}\}\{\textit{false}\}
\hspace{1em} initial: \textit{false}\hspace{1em}

User hook to trigger a check for a following full stop. This should do \textit{true} if there should be a check for a following full stop otherwise should do \textit{false}.

\glxtrifemptyglossary\{\textit{glossary-type}\}\{\textit{true}\}\{\textit{false}\}

Does \textit{true} if the glossary identified by \textit{glossary-type} is empty, otherwise does \textit{false}. If the glossary doesn’t exist, this does \textit{true} and will either generate an error (\texttt{undefaction=error}) or a warning (\texttt{undefaction=warn}). This command considers ignored glossaries as existing.

\GlsXtrIfFieldCmpNum\{\textit{field-label}\}\{\textit{entry-label}\}\{\textit{op}\}\{\textit{number}\}\{\textit{true}\}\{\textit{false}\}
\hspace{1em} modifier: * \hspace{1em}

Compares the (numeric) value of the field identified by its internal label \textit{field-label} for the entry identified by \textit{entry-label} with \textit{number} where \textit{op} is the comparison operator (\texttt{=}, \texttt{<}, or \texttt{>}). The unstarred version adds implicit grouping. The starred version doesn’t.

\GlsXtrIfFieldEqNum\{\textit{field-label}\}\{\textit{entry-label}\}\{\textit{number}\}\{\textit{true}\}\{\textit{false}\}
\hspace{1em} modifier: * \hspace{1em}

A shortcut that uses \GlsXtrIfFieldCmpNum with \textit{op} set to \texttt{=} \texttt{=}. The unstarred version adds implicit grouping. The starred version doesn’t.

\GlsXtrIfFieldEqStr\{\textit{field-label}\}\{\textit{entry-label}\}\{\textit{value}\}\{\textit{true}\}\{\textit{false}\}
\hspace{1em} modifier: * \hspace{1em}

Tests if the entry given by \textit{entry-label} has the field identified by its internal label \textit{field-label} set to \textit{value}. This internally uses \texttt{\glxtrifhasfield} and compares \texttt{\glscurrent-...
fieldvalue to \langle value \rangle using etoolbox's \ifdefstring. The unstarred version adds implicit grouping. The starred version doesn't.

\texttt{\GlsXtrIfFieldEqXpStr{\langle field-label \rangle}{\langle entry-label \rangle}{\langle value \rangle}{\langle true \rangle}{\langle false \rangle}}

\texttt{\GlsXtrIfFieldEqStr{\langle field-label \rangle}{\langle entry-label \rangle}{\langle value \rangle}{\langle true \rangle}{\langle false \rangle}}

Like \texttt{\GlsXtrIfFieldEqStr} but first (protected) expands \langle value \rangle.

\texttt{\GlsXtrIfFieldNonZero{\langle field-label \rangle}{\langle entry-label \rangle}{\langle true \rangle}{\langle false \rangle}}

A shortcut that uses \texttt{\GlsXtrIfFieldCmpNum} to test if the (numeric) value of the field identified by its internal label \langle field-label \rangle for the entry identified by \langle entry-label \rangle is non-zero. An empty or undefined field is treated as 0. The unstarred version adds implicit grouping. The starred version doesn't. The value can be referenced within \langle true \rangle (where it will be 0) or within \langle false \rangle using \texttt{\glscurrentfieldvalue}.

\texttt{\GlsXtrIfFieldUndef{\langle field-label \rangle}{\langle entry-label \rangle}{\langle true \rangle}{\langle false \rangle}}

Expandable command that tests if the given field (identified by its internal label) is undefined for the entry given by \langle entry-label \rangle. Internally uses etoolbox's \ifdefcsundef command. Unlike \texttt{\glsxtrifhasfield} there is no grouping or starred version.

\texttt{\GlsXtrIfFieldValueInCsvList{\langle entry-label \rangle}{\langle field-label \rangle}{\langle csv-list \rangle}{\langle true \rangle}{\langle false \rangle}}

Tests if the value stored in the given field (identified by its internal label) for the entry identified by \langle entry-label \rangle is contained in the comma-separated list \langle csv-list \rangle using \texttt{\DTLifinlist} (provided by datatool-base, which is automatically loaded by the glossaries package). One level expansion is performed on \langle csv-list \rangle. This command uses \texttt{\glsxtrifhasfield} so the field value can be obtained with \texttt{\glscurrentfieldvalue}. The unstarred version adds implicit grouping. The starred version doesn't.

\texttt{\glsxtrifhasfield{\langle field-label \rangle}{\langle entry-label \rangle}{\langle true \rangle}{\langle false \rangle}}

\texttt{\glsxtrifhasfield{\langle field-label \rangle}{\langle entry-label \rangle}{\langle true \rangle}{\langle false \rangle}}

819
Command Summary

Tests if the field identified by its internal label \(\langle field-label \rangle\) for the entry given by \(\langle entry-label \rangle\) is defined and is not empty. This is like \texttt{ifgls\textbackslash hasfield} but doesn't produce a warning if the entry or field doesn't exist. This sets \texttt{\glscurrentfieldvalue} to the field value and does \(\langle true \rangle\) if its defined and not empty, otherwise it does \(\langle false \rangle\). The unstarred version adds implicit grouping to make nesting easier. The starred version doesn't (to make assignments easier).

\[
\GlsXtrIfHasNonZeroChildCount\{\langle entry-label \rangle\}\{\langle true \rangle\}\{\langle false \rangle\} \quad \text{modifier: } *
\]
glossaries-extra-bib2gls v1.47+

Tests if the value in the \texttt{childcount} field is non-zero (using \texttt{\GlsXtrIfFieldNonZero}). This requires the \texttt{save-child-count} resource option.

\[
\texttt{\glsxtrifheaduc}\{\langle entry-label \rangle\}\{\langle true \rangle\}\{\langle false \rangle\} \quad \text{glossaries-extra v1.49+}
\]

If the category associated with the entry given by \(\langle entry-label \rangle\) has the \texttt{headuc} attribute set to \texttt{true} this does \(\langle true \rangle\) otherwise it does \(\langle false \rangle\).

\[
\texttt{\glsxtrifhyphenstart}\{\langle text \rangle\}\{\langle true \rangle\}\{\langle false \rangle\} \quad \text{glossaries-extra v1.17+}
\]

If \(\langle text \rangle\) starts with a hyphen this does \(\langle true \rangle\) otherwise it does \(\langle false \rangle\).

\[
\texttt{\glsxtrifindexing}\{\langle true \rangle\}\{\langle false \rangle\} \quad \text{glossaries-extra v1.49+}
\]

Tests whether or not the \texttt{noindex} has been set. Does \(\langle false \rangle\) if \texttt{noindex=true} otherwise does \(\langle true \rangle\).

\[
\texttt{\GlsXtrIfInGlossary}\{\langle entry-label \rangle\}\{\langle glossary-type \rangle\}\{\langle true \rangle\}\{\langle false \rangle\} \quad \text{glossaries-extra v1.49+}
\]

Does \(\langle true \rangle\) if the entry given by \(\langle entry-label \rangle\) is in the internal list of the glossary identified by \(\langle glossary-type \rangle\), otherwise it does \(\langle false \rangle\). If the glossary doesn't exist, this does \(\langle false \rangle\) and will either generate an error (\texttt{undefaction=error}) or a warning (\texttt{undefaction=warn}). This command considers ignored glossaries as existing.
Command Summary

\glsxtrifinlabelprefixlist\{\langle label-prefix\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra-bib2gls v1.37+

§11.5.7; 563

Does \langle true\rangle if \langle label-prefix\rangle has been identified as a label prefix.

\glsxtrifinmark\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra v1.07+

§5.3.3; 214

Does \langle true\rangle if within \markright, \markboth or \@starttoc otherwise does \langle false\rangle.

\glsxtrifintoc\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra v1.49+

§5.3.3; 214

Normally just expands to \langle false\rangle but \@starttoc is redefined to temporarily set this macro to expand to \langle true\rangle instead. Will always expand to \langle false\rangle if \glsxtrRevertTocMarks or \glsxtrRevertMarks are used to revert \@starttoc to its former definition.

\glsxtrifkeydefined\{\langle key\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra v1.12+

§3.2; 35

Tests if the given \langle key\rangle has been defined as a glossary entry key. Does \langle true\rangle if the key has been defined, otherwise does false.

\glsxtriflabelinlist\{\langle label\rangle\}\{\langle label-list\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra v1.21+

§8.4.3; 402

Does \langle true\rangle if the given label is in the given comma-separated list of labels, otherwise does \langle false\rangle. The label and list are fully expanded.

\GlsXtrIfLinkCounterDef\{\langle entry-label\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra v1.26+

§6.2; 318

Expands to \langle true\rangle if the link counter associated with the given entry has been defined, otherwise expands to \langle false\rangle.

\glsxtrifmulti\{\langle multi-label\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}
glossaries-extra v1.48+

§7.13; 370
Does \texttt{⟨true⟩} if a multi-entry has been defined with the label \texttt{⟨multi-label⟩} otherwise does \texttt{⟨false⟩}.

\begin{verbatim}
\glsxtrifnextpunc{⟨true⟩}{⟨false⟩}
glossaries-extra
\end{verbatim}

Performs \texttt{⟨true⟩} if this command is followed by a recognised punctuation character otherwise does \texttt{⟨false⟩}. The list of recognised punctuation marks is initialised to . , ; ; ? ! (full stop, comma, colon, semicolon, question mark, and exclamation mark). Additional punctuation characters can be added with \glsxtraddpunctuationmark.

\begin{verbatim}
\glsxtrifperiod{⟨true⟩}{⟨false⟩}{⟨token⟩}
glossaries-extra
\end{verbatim}

Does \texttt{⟨true⟩} if \texttt{⟨token⟩} is a full stop, otherwise does \texttt{⟨false⟩}.

\begin{verbatim}
\glsxtrifrecordtrigger{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
glossaries-extra v1.21+
\end{verbatim}

Does \texttt{⟨true⟩} if the entry’s total record count (obtained with \glsxtrrecordtriggervalue) exceeds the value supplied by the \texttt{recordcount} attribute, otherwise does \texttt{⟨false⟩}.

\begin{verbatim}
\GlsXtrIfUnusedOrUndefined{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
glossaries-extra v1.34+
\end{verbatim}

Does \texttt{⟨true⟩} if the entry hasn’t been defined or hasn’t been marked as used, otherwise does \texttt{⟨true⟩}. Note that this command will generate an error or warning (according to \texttt{undef-action}) if the entry hasn’t been defined, but will still do \texttt{⟨true⟩}.

\begin{verbatim}
\GlsXtrIfValueInFieldCsvList{⟨entry-label⟩}{⟨field-label⟩}{⟨value⟩}{⟨true⟩}{⟨false⟩}
modifier: *
\end{verbatim}

Tests if the given value (⟨value⟩) is contained in the comma-separated list stored in the given field (identified by its internal label) for the entry identified by \texttt{⟨entry-label⟩} using \texttt{\DTLifin-list} (provided by datatool-base, which is automatically loaded by the glossaries package). No expansion is performed on ⟨value⟩. This command uses \glsxtrifhasfield so the
complete list can be obtained with \gls{currentfieldvalue}. The unstarred version adds implicit grouping. The starred version doesn’t.

\glsxtrifwasfirstuse{⟨true⟩}{⟨false⟩} §5.10; 282

Initialised by the \gls-like and \glstext-like commands, this expands to ⟨true⟩ if the calling command was considered the first use, otherwise it expands to ⟨false⟩. This command may be used within the post-link hook (where it’s too late to test the first use flag with \ifglsused).

\glsxtrifwasglslike{⟨true⟩}{⟨false⟩} glossaries-extra v1.49+ §5.5.4; 252

Initialised by the \gls-like and \glstext-like commands, this expands to ⟨true⟩ if the calling command was a \gls-like command, otherwise it expands to ⟨false⟩. This command may be used within the post-link hook.

\glsxtrifwasglslikeandfirstuse{⟨true⟩}{⟨false⟩} glossaries-extra v1.49+ §5.5.4; 252

A shortcut that nests \glsxtrifwasglslike and \glsxtrifwasfirstuse. This does ⟨true⟩ if the calling command was both a \gls-like command and was considered the first use.

\glsxtrifwassubsequentorshort{⟨true⟩}{⟨false⟩} glossaries-extra v1.49+ §5.5.4; 253

Expands to ⟨true⟩ if the calling command was a \gls-like command and was the subsequent use or if \glsxtrcurrentfield was set to short.

\glsxtrifwassubsequentuse{⟨true⟩}{⟨false⟩} glossaries-extra v1.49+ §5.5.4; 252

A shortcut that nests \glsxtrifwasglslike and \glsxtrifwasfirstuse. This does ⟨true⟩ if the calling command was a \gls-like command but was not considered the first use.

\GlsIfXpFieldEqXpStr{⟨field-label⟩}{⟨entry-label⟩}{⟨value⟩}{⟨true⟩}{{false}} modifier: * glossaries-extra v1.31+ §5.15; 306

823
Command Summary

Like \GlsXtrIfFieldEqStr but first (protected) expands both the field value and the supplied \textit{value}.

\texttt{\glsxtrIgnorableRules} \hfill glossaries-extra-bib2gls v1.49+

A shortcut that expands to the control rules, space rules and non-printable rules.

\texttt{\glsxtrlinclinkcounter\{\textit{counter}\}} \hfill glossaries-extra v1.26+

Increments the link counter with \texttt{\textbackslash stepcounter}.

\texttt{\glsxtrindexaliased} \hfill glossaries-extra v1.12+

Index the current entry’s alias. May only be used within the definition of \texttt{\glsxtrsetalias-noindex}.

\texttt{\GlsXtrIndexCounterLink\{\textit{text}\}\{\textit{entry-label}\}} \hfill glossaries-extra-bib2gls v1.29+

Creates a hyperlink (if supported) to the target obtained from \texttt{indexcounter}, if the field has been defined with the given hyperlink text (otherwise just does \texttt{\langle text\rangle}).

\texttt{\glsxtrindexseealso\{\textit{entry-label}\}\{\textit{xr-list}\}} \hfill glossaries-extra v1.16+

Indexes the entry identified by \texttt{\langle entry-label\rangle} as a “see also” cross-reference to the entries identified in the comma-separated list \texttt{\langle xr-list\rangle}. The cross-reference list is prefixed with \texttt{\langle see-also-name\rangle}.

\texttt{\glsxtrinihyperoutside} \hfill glossaries-extra v1.21+

Hook that initialises the \texttt{hyperoutside} setting.
Command Summary

\glsxtrinitwrgloss  
glossaries-extra v1.14+ §5.1.1; 189

Hook that initialises the \texttt{wrgloss} setting.

\glsxtrinitwrglossbeforefalse  
glossaries-extra v1.14+

Corresponds to \texttt{wrgloss=after}.

\glsxtrinitwrglossbeforetrue  
glossaries-extra v1.14+

Corresponds to \texttt{wrgloss=before}.

\GLSxtrinlinefullformat\{⟨entry-label⟩\}{⟨insert⟩}  
glossaries-extra v1.49+

175

Used by \texttt{\GLSxtrfull} to display the all caps inline full form form (defined by the abbreviation style).

\Glsxtrinlinefullformat\{⟨entry-label⟩\}{⟨insert⟩}

175

Used by \texttt{\Glsxtrfull} to display the sentence case inline full form form (defined by the abbreviation style).

\glxtrinlinefullformat\{⟨entry-label⟩\}{⟨insert⟩}

175

Used by \texttt{\glxtrfull} to display the inline full form form (defined by the abbreviation style).

\GLSxtrinlinefullplformat\{⟨entry-label⟩\}{⟨insert⟩}  
glossaries-extra v1.49+

175

Used by \texttt{\GLSxtrfullpl} to display the plural all caps inline full form form (defined by the abbreviation style).
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\Glsxtrinlinefullplformat{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Used by \Glsxtrfullpl to display the plural sentence case inline full form (defined by the abbreviation style).</td>
</tr>
<tr>
<td>\glsxtrinlinefullplformat{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Used by \glsxtrfullpl to display the plural inline full form form (defined by the abbreviation style).</td>
</tr>
<tr>
<td>\glsxtrinsertinsidefalse</td>
<td>Sets the \ifglsxtrinsertinside conditional to false.</td>
</tr>
<tr>
<td>\glsxtrinsertinsidetrue</td>
<td>Sets the \ifglsxtrinsertinside to true.</td>
</tr>
<tr>
<td>\GlsXtrInternalLocationHyperlink{⟨counter⟩}{⟨prefix⟩}{⟨location⟩}</td>
<td>Used by \glsxtrlocationhyperlink to create an internal hyperlink to the given location (advanced command, see documented code for use).</td>
</tr>
<tr>
<td>\glsxtrLatinA</td>
<td>(Sort rule) expands to the variations of Latin A (includes 0x00AA and 0x2090).</td>
</tr>
<tr>
<td>\glsxtrLatinAA</td>
<td>(Sort rule) expands to the variations of Latin à.</td>
</tr>
</tbody>
</table>
Command Summary

\texttt{\textbackslash glsxtrLatinAE\textsubscript{Ligature}} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of Latin ae-ligature.

\texttt{\textbackslash glsxtrLatinE} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of Latin E (includes 0x2091).

\texttt{\textbackslash glsxtrLatinEszet\textsubscript{t}Ss} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to rule for ũ and ſ.

\texttt{\textbackslash glsxtrLatinEszet\textsubscript{t}Sz} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to rule for ũ and ſz.

\texttt{\textbackslash glsxtrLatinE\textsubscript{th}}} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of Latin eth.

\texttt{\textbackslash glsxtrLatinH} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of Latin H (includes 0x2095).

\texttt{\textbackslash glsxtrLatinI} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of Latin I (includes 0x2071).

\texttt{\textbackslash glsxtrLatinInsularG} \quad \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of Latin insular G and g, G.

827
## Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{\texttt{_glsxtrLatinK}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin K (includes 0x2096).</td>
</tr>
<tr>
<td>\textit{\texttt{_glsxtrLatinL}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin L (includes 0x2097).</td>
</tr>
<tr>
<td>\textit{\texttt{_glsxtrLatinLslash}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin ł.</td>
</tr>
<tr>
<td>\textit{\texttt{_glsxtrLatinM}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin M (includes 0x2098).</td>
</tr>
<tr>
<td>\textit{\texttt{_glsxtrLatinN}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin N (includes 0x2099).</td>
</tr>
<tr>
<td>\textit{\texttt{_glsxtrLatinO}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin O (includes 0x00BA and 0x2092).</td>
</tr>
<tr>
<td>\textit{\texttt{_glsxtrLatinOELigature}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin oe-ligature.</td>
</tr>
<tr>
<td>\textit{\texttt{_glsxtrLatinOslash}}</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>(Sort rule)</td>
<td>expands to the variations of Latin ø.</td>
</tr>
</tbody>
</table>
\textbf{Command Summary}

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrLatinP</td>
<td>(Sort rule) expands to the variations of Latin P (includes 0x209A).</td>
</tr>
<tr>
<td>\glsxtrLatinS</td>
<td>(Sort rule) expands to the variations of Latin S (includes 0x209B).</td>
</tr>
<tr>
<td>\glsxtrLatinSchwa</td>
<td>(Sort rule) expands to the variations of Latin schwa.</td>
</tr>
<tr>
<td>\glsxtrLatinT</td>
<td>(Sort rule) expands to the variations of Latin T (includes 0x209C).</td>
</tr>
<tr>
<td>\glsxtrLatinThorn</td>
<td>(Sort rule) expands to the variations of Latin thorn.</td>
</tr>
<tr>
<td>\glsxtrLatinWynn</td>
<td>(Sort rule) expands to the variations of Latin wynn.</td>
</tr>
<tr>
<td>\glsxtrLatinX</td>
<td>(Sort rule) expands to the variations of Latin X (includes 0x2093).</td>
</tr>
</tbody>
</table>

Like \texttt{\GlsXtrLetField} but internally uses (etoolbox’s) \texttt{\cslet} instead of \texttt{\csdef}.

§3.5; 40

829
Assigns the field identified by its internal label \(\langle field1\text{-}label\rangle\) for the entry identified by \(\langle entry1\text{-}label\rangle\) to the value of the field identified by \(\langle field2\text{-}label\rangle\) for the entry identified by \(\langle entry2\text{-}label\rangle\).

Expands to the name of the link counter associated with the given entry (no check for existence).

Expands to the internal link count register associated with the given register or 0 if it hasn’t been defined.

A shortcut that uses \glsxtrresourcefile\(\langle options\rangle\)\(\langle basename\rangle\), where the \(\langle basename\rangle\) is obtained from \jobname and \glsxtrresourcecount.

Locally assigns the given title \(\langle group\text{-}title\rangle\) to the group identified by \(\langle group\text{-}label\rangle\).

Defined by \glsxtrdisplaylocnameref to expand to the anchor constructed from \(\langle counter\rangle\) and \(\langle hcounter\rangle\), which corresponds to the record counter.
Command Summary

Expands to the internal field label used to obtain the formatted location list for the “unsrt” family of commands.

\(\text{\texttt{\textbackslash glsxtroperationhyperlink\{\langle counter\rangle\{\langle prefix\rangle\{\langle location\rangle\}}}\text{\textbackslash glossaries-extra v1.14+}}\)

Used to create a hyperlink to either an external or an internal location, depending on whether or not \(\text{\texttt{\textbackslash glsxtroperationlocationurl}}\) is defined and not empty (advanced command, see documented code for use).

\(\text{\texttt{\textbackslash GlsXtrLocationRecordCount\{\langle entry-label\rangle\{\langle counter\rangle\{\langle location\rangle\}}}\text{\textbackslash glossaries-extra v1.21+}}\)

Expands to the entry’s record count for the given counter and location (stored in the record-count.\langle counter\rangle.\langle location\rangle field) or to 0 if not set.

\(\text{\texttt{\textbackslash glsxtroperationrangefmt}}\text{\textbackslash glossaries-extra v1.12+}\)

Expands to the range format (set by \(\text{\texttt{\textbackslash glsxtroperationdisplaystartloc}}\)).

\(\text{\texttt{\textbackslash GLSxtrlong[\langle options\rangle]{\langle entry-label\rangle}[\langle insert\rangle] modifiers: * + \langle alt-mod\rangle}}\text{\textbackslash §4.3; 53}\)

As \(\text{\texttt{\textbackslash glsxtroperationlong}}\) but converts the link text to all caps.

\(\text{\texttt{\textbackslash Glsxtrlong[\langle options\rangle]{\langle entry-label\rangle}[\langle insert\rangle] modifiers: * + \langle alt-mod\rangle}}\text{\textbackslash §4.3; 53}\)

As \(\text{\texttt{\textbackslash glsxtroperationlong}}\) but converts the first character of the link text to uppercase (for the start of a sentence) using \(\text{\texttt{\textbackslash makefirstuc}}\).

\(\text{\texttt{\textbackslash glsxtroperationlong[\langle options\rangle]{\langle entry-label\rangle}[\langle insert\rangle] modifiers: * + \langle alt-mod\rangle}}\text{\textbackslash §4.3; 52}\)

References the entry identified by \(\langle entry-label\rangle\). The text produced is obtained from the long value, formatted according to the abbreviation style associated with the entry’s category. The
Command Summary

The \langle insert\rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \glslink options.

\GLSxtrlongformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\} glossaries-extra v1.49+

As \glsxtrlongformat but sentence case.

\Glsxtrlongformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\} glossaries-extra v1.49+

As \glsxtrlongformat but sentence case.

\glsxtrlongformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\} glossaries-extra v1.49+

Encapsulates the long field for the given entry with \langle fmt-cs\rangle. The \langle insert\rangle argument is the insertion material supplied in the final optional argument of the \glslike or \glstextlike commands. The \if\glsxtrinsertinside, inner formatting, and accessibility settings are supported.

\GLSxtrlongformatgrp\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\} glossaries-extra v1.49+

As \glsxtrlongformatgrp but all caps.

\Glsxtrlongformatgrp\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\} glossaries-extra v1.49+

As \glsxtrlongformatgrp but sentence case.

\glsxtrlongformatgrp\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\} glossaries-extra v1.49+

As \glsxtrlongformat but adds grouping around \langle insert\rangle (with the inner formatting inside the group).

\glsxtrlonghyphen\{\langle entry-label\rangle\}\{\langle long\rangle\}\{\langle insert\rangle\} glossaries-extra v1.17+
Command Summary

Formats the long form according to the \texttt{long-hyphen-postshort-hyphen} style.

\GLSxtrlonghyphennoshort\{⟨entry-label⟩\}{⟨long⟩}{⟨insert⟩}\glossaries-extra v1.49+

As \texttt{\GLSxtrlonghyphennoshort} but converts ⟨insert⟩ to all caps. The ⟨long⟩ argument should be supplied as all caps.

\GLSxtrlonghyphennoshort\{⟨entry-label⟩\}{⟨long⟩}{⟨insert⟩}\glossaries-extra v1.17+

Formats the long form according to the \texttt{long-hyphen-noshort-desc-noreg} style.

\GLSxtrlonghyphennoshortdescsort \glossaries-extra v1.49+

Expands to the sort value for the \texttt{long-hyphen-noshort-desc-noreg} styles.

\GLSxtrlonghyphennoshortsort \glossaries-extra v1.49+

Expands to the sort value for the \texttt{long-hyphen-noshort-noreg} styles.

\GLSxtrlonghyphenshort\{⟨entry-label⟩\}{⟨long⟩}{⟨short⟩}{⟨insert⟩}\glossaries-extra v1.49+

As \texttt{\GLSxtrlonghyphenshort} but converts ⟨insert⟩ to all caps. The ⟨long⟩ and ⟨short⟩ arguments should be supplied as all caps.

\GLSxtrlonghyphenshort\{⟨entry-label⟩\}{⟨long⟩}{⟨short⟩}{⟨insert⟩}\glossaries-extra v1.17+

Formats the long and short form according to the \texttt{long-hyphen-short-hyphen} style.

\GLSxtrlonghyphenshortsort \glossaries-extra v1.49+
Command Summary

Expands to the sort value for the `long-hyphen-short-hyphen` styles.

\glsxtrlongnoshortdesctextname \hspace{1em} glossaries-extra v1.25+

Expands to the name value for styles like `long-noshort-desc`.

\glsxtrlongnoshortname \hspace{1em} glossaries-extra v1.25+

Expands to the name value for styles like `long-noshort`.

\GLSxtrlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} \hspace{1em} modifiers: * + ⟨alt-mod⟩

§4.3; 53

As `\glsxtrlongpl` but converts the link text to all caps.

\GLSxtrlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} \hspace{1em} modifiers: * + ⟨alt-mod⟩

§4.3; 53

As `\glsxtrlongpl` but converts the first character of the link text to uppercase (for the start of a sentence) using `\makefirstuc`.

\GLSxtrlongpl[⟨options⟩]{⟨entry-label⟩}{⟨insert⟩} \hspace{1em} modifiers: * + ⟨alt-mod⟩

§4.3; 53

References the entry identified by `⟨entry-label⟩`. The text produced is obtained from the `longplural` value, formatted according to the abbreviation style associated with the entry’s category. The `⟨insert⟩` argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see `\glslink` options.

\GLSxtrlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩} \hspace{1em} glossaries-extra v1.49+

As `\glsxtrlongplformat` but all caps.
\Glsxtrlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrlongformat but sentence case.

\glsxtrlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrlongformat but for the \texttt{longplural} field.

\GLSxtrlongplformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrlongplformatgrp but all caps.

\Glsxtrlongplformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrlongplformatgrp but sentence case.

\glsxtrlongplformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrlongplformatgrp but adds grouping around \langle insert \rangle (with the inner formatting inside the group).

\glsxtrlongshortdescname
glossaries-extra v1.17+

Expands to the name value for \texttt{long-short-desc} styles.

\glsxtrlongshortdescsort
glossaries-extra v1.04+

Expands to the sort value for \texttt{long-short-desc} styles.

\GLSxtrlongshortformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+
Command Summary

As \glsxtrlongshortformat but all caps.

\begin{verbatim}
\Glsxtrlongshortformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+
\end{verbatim}

As \glsxtrlongshortformat but sentence case.

\begin{verbatim}
\glsxtrlongshortformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+
\end{verbatim}

Formats the long form with \glsxtrlongformat and the short form in parentheses with \glsxtrshortformat.

\begin{verbatim}
\glsxtrlongshortname
glossaries-extra v1.25+
\end{verbatim}

Expands to the name value for long–short styles.

\begin{verbatim}
\GLSxtrlongshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+
\end{verbatim}

As \glsxtrlongshortplformat but all caps.

\begin{verbatim}
\Glsxtrlongshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+
\end{verbatim}

As \glsxtrlongshortplformat but sentence case.

\begin{verbatim}
\glsxtrlongshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+
\end{verbatim}

As \glsxtrlongshortformat but for the plurals.
Command Summary

\glsxtrlongshortscuserdescname \textit{glossaries-extra v1.48+}
Expands to the value for the name key for styles like long-postshort-sc-user-desc.

\glsxtrlongshortscusername \textit{glossaries-extra v1.48+}
Expands to the value for the name key for styles like long-postshort-sc-user.

\glsxtrlongshortuserdescname \textit{glossaries-extra v1.25+}
Expands to the value for the name key for styles like long-short-user-desc.

\glsxtrmarkhook \textit{§5.3.3; 227}
Hook that's performed at the start of \markright, \markboth and \@starttoc to redefine commands that need to change when they occur within page headers or contents. This must be counteracted with \glsxtrrestoremarkhook afterwards.

\glsxtrMathGreekIIrules \textit{glossaries-extra-bib2gls v1.27+}
Expands to the second set of math Greek sort rules.

\glsxtrMathGreekIrules \textit{glossaries-extra-bib2gls v1.27+}
Expands to the first set of math Greek sort rules.

\glsxtrMathItalicAlpha \textit{glossaries-extra-bib2gls v1.27+}
(Sort rule) expands to the variations of math italic Greek alpha.

\glsxtrMathItalicBeta \textit{glossaries-extra-bib2gls v1.27+}
(Sort rule) expands to the variations of math italic Greek beta.

\texttt{\glsxtrMathItalicChi} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math italic Greek chi.

\texttt{\glsxtrMathItalicDelta} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math italic Greek delta.

\texttt{\glsxtrMathItalicDigamma} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math italic Greek digamma.

\texttt{\glsxtrMathItalicEpsilon} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math italic Greek epsilon.

\texttt{\glsxtrMathItalicEta} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math italic Greek eta.

\texttt{\glsxtrMathItalicGamma} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math italic Greek gamma.

\texttt{\glsxtrMathItalicGreekIIrules} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

Expands to the second set of math italic Greek sort rules.
### Command Summary

<table>
<thead>
<tr>
<th>Command Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrMathItalicGreekIrules</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>\glsxtrMathItalicIota</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>\glsxtrMathItalicKappa</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>\glsxtrMathItalicLambda</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>\glsxtrMathItalicLowerGreekIIrules</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>\glsxtrMathItalicLowerGreekIrules</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>\glsxtrMathItalicMu</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
<tr>
<td>\glsxtrMathItalicNabla</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
</tbody>
</table>
### Command Summary

**\glsxtrMathItalicNu**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the variations of math italic Greek nu.

**\glsxtrMathItalicOmega**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the variations of math italic Greek omega.

**\glsxtrMathItalicOmicron**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the variations of math italic Greek omicron.

**\glsxtrMathItalicPartial**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the Unicode codepoint for math italic partial differential.

**\glsxtrMathItalicPhi**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the variations of math italic Greek phi.

**\glsxtrMathItalicPi**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the variations of math italic Greek pi.

**\glsxtrMathItalicPsi**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the variations of math italic Greek psi.

**\glsxtrMathItalicRho**  
glossaries-extra-bib2gls v1.27+

(Sort rule) expands to the variations of math italic Greek rho.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glsxtrMathItalicSigma</code></td>
<td>(Sort rule) expands to the variations of math italic Greek sigma.</td>
</tr>
<tr>
<td><code>\glsxtrMathItalicTau</code></td>
<td>(Sort rule) expands to the variations of math italic Greek tau.</td>
</tr>
<tr>
<td><code>\glsxtrMathItalicTheta</code></td>
<td>(Sort rule) expands to the variations of math italic Greek theta.</td>
</tr>
<tr>
<td><code>\glsxtrMathItalicUpperGreekIIRules</code></td>
<td>Expands to the second set of math italic uppercase Greek sort rules.</td>
</tr>
<tr>
<td><code>\glsxtrMathItalicUpperGreekIIRules</code></td>
<td>Expands to the first set of math italic uppercase Greek sort rules.</td>
</tr>
<tr>
<td><code>\glsxtrMathItalicUpsilon</code></td>
<td>(Sort rule) expands to the variations of math italic Greek upsilon.</td>
</tr>
<tr>
<td><code>\glsxtrMathItalicXi</code></td>
<td>(Sort rule) expands to the variations of math italic Greek xi.</td>
</tr>
<tr>
<td><code>\glsxtrMathItalicZeta</code></td>
<td>(Sort rule) expands to the variations of math italic Greek zeta.</td>
</tr>
<tr>
<td>Command Summary</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>\glsxtrMathUpGreekIIrules</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
</tbody>
</table>

Expands to the second set of math upright Greek sort rules.

| \glsxtrMathUpGreekIrules | glossaries-extra-bib2gls v1.27+ |

Expands to the first set of math upright Greek sort rules.

| \glsxtrMFUsave | glossaries-extra v1.49+ |

Used on the first instance of \glsxtrresourcefile, this will add \MFUsave to the begin document hook and then disable itself. This is provided to help bib2gls pick up any of mfirstuc’s exclusions, blockers and mappings to assist with its sentence case function.

| \GlsXtrMglsOrGls \{\langle mgls cs\rangle\}\{\langle gls cs\rangle\}\{\langle modifier\rangle [\langle options\rangle] [\langle label\rangle] [\langle insert\rangle] \} | glossaries-extra v1.48+ |

If \langle label\rangle matches a defined multi-entry, this will do \langle mgls cls\rangle otherwise it will do \langle gls cs\rangle. The \langle modifier\rangle (∗ or + or the token identified with \GlsXtrSetAltModifier) may be omitted.

| \glsxtrmglsWarnAllSkipped \{\langle message\rangle\}\{\langle insert\rangle\}\{\langle fmt-cs\rangle\} | glossaries-extra v1.48+ |

Issues the given warning message with \GlossariesExtraWarning and does \langle fmt-cs\rangle\{\langle insert\rangle\} (this warning is used if all elements of a multi-entry set are skipped).

| \GLSxtrmultientryadjustedname \{\langle sublist\rangle\}\{\langle name\rangle\}\{\langle sublist2\rangle\}\{\langle multi-label\rangle\} | glossaries-extra-bib2gls v1.48+ |

As \glsxtrmultientryadjustedname but all caps.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\GlsXtrmultientryadjustedname{(sublist1)}{(name)}{(sublist2)}{(multi-label)}</td>
<td>As \glsxtrmultientryadjustedname but title case.</td>
</tr>
<tr>
<td>\GLSxtrmultientryadjustednamefmt{(text)}</td>
<td>Used by \GLSxtrmultientryadjustedname to encapsulate the main entry name.</td>
</tr>
<tr>
<td>\GlsXtrmultientryadjustednamefmt{(text)}</td>
<td>Used by \GlsXtrmultientryadjustedname to encapsulate the main entry name if the first sublist is empty.</td>
</tr>
</tbody>
</table>

843
Command Summary

\GLSxtrmultientryadjustednameother{⟨text⟩} glossaries-extra-bib2gls v1.48+

§7.14; 374

Used by \GLSxtrmultientryadjustedname to encapsulate the other (not main) entries.

\GlsXtrmultientryadjustednameother{⟨text⟩} glossaries-extra-bib2gls v1.48+

§7.14; 374

Used by \Glsxtrmultientryadjustedname to encapsulate the other (not main) entries.

\Glsxtrmultientryadjustednameother{⟨text⟩} glossaries-extra-bib2gls v1.48+

§7.14; 374

Used by \Glsxtrmultientryadjustedname to encapsulate the other (not main) entries.

\glsxtrmultientryadjustednameother{⟨text⟩} glossaries-extra-bib2gls v1.48+

§7.14; 374

Used by \glsxtrmultientryadjustedname to encapsulate the other (not main) entries.

\glsxtrmultientryadjustednamepostsep{⟨pre-label⟩}{⟨post-label⟩}

glossaries-extra-bib2gls v1.48+

§7.14; 373

Separator used by \glsxtrmultientryadjustedname between the main element and the first element of the second sublist.

\glsxtrmultientryadjustednamepresep{⟨pre-label⟩}{⟨post-label⟩}

glossaries-extra-bib2gls v1.48+

§7.14; 373

Separator used by \glsxtrmultientryadjustedname between the last element of the first sublist and the main element.

\glsxtrmultientryadjustednamesep{⟨pre-label⟩}{⟨post-label⟩}

glossaries-extra-bib2gls v1.48+

§7.14; 373

Separator used by \glsxtrmultientryadjustedname.
Command Summary

\texttt{\glsxtrmultilastotherindex\{\textit{multi-label}\}} \quad \text{glossaries-extra v1.48+} \quad \S 7.13; 371

Expands to the index of the final non-main element in the given multi-entry or nothing if \textit{\texttt{multi-label}} hasn’t been defined.

\texttt{\glsxtrmultilist\{\textit{multi-label}\}} \quad \text{glossaries-extra v1.48+} \quad \S 7.13; 370

Expands to the list of element labels for the multi-entry identified by \textit{\texttt{multi-label}} or nothing if not defined.

\texttt{\glsxtrmultimain\{\textit{multi-label}\}} \quad \text{glossaries-extra v1.48+} \quad \S 7.13; 370

Expands to the main label for the multi-entry identified by \textit{\texttt{multi-label}} or nothing if not defined.

\texttt{\glsxtrmultimainindex\{\textit{multi-label}\}} \quad \text{glossaries-extra v1.48+} \quad \S 7.13; 371

Expands to the index of the main element in the given multi-entry or nothing if \textit{\texttt{multi-label}} hasn’t been defined.

\texttt{\glsxtrmultisupplocation\{\textit{src}\}\{\textit{location}\}\{\textit{format}\}} \quad \text{modifier: *} \quad \text{glossaries-extra-bib2gls v1.36+} \quad \S 11.5.5; 555

Used by \texttt{\glsxtrdisplaysupploc} to format the location.

\texttt{\glsxtrmultitotalelements\{\textit{multi-label}\}} \quad \text{glossaries-extra v1.48+} \quad \S 7.13; 371

Expands to the total number of elements in the given multi-entry or nothing if \textit{\texttt{multi-label}} hasn’t been defined.

\texttt{\glsxtrnameloclink\{\textit{prefix}\}\{\textit{counter}\}\{\textit{format}\}\{\textit{location}\}\{\textit{text}\}\{\textit{file}\}} \quad \text{glossaries-extra-bib2gls v1.37+} \quad \S 11.5.6; 560
Create an external location hyperlink using the prefix and counter.

\glsxtrnameref{⟨format⟩}{⟨title⟩}{⟨target⟩}{⟨file⟩}
glossaries-extra-bib2gls v1.37+

Used by \glsxtrdisplaylocnameref to create a location hyperlink.

\glsxtrnewabbrevpresetkeyhook{⟨options⟩}{⟨label⟩}{⟨short⟩}
§4.1.5; 45

Hook provided to adjust initialisation within \newabbreviation.

\glsxtrnewgls[⟨default-options⟩]{⟨prefix⟩}{⟨cs⟩}  glossaries-extra v1.21+
§5.7; 257

Defines the command \{⟨options⟩\}{⟨entry-label⟩} to behave like \gls[⟨default-options⟩,⟨options⟩]{⟨prefix⟩⟨entry-label⟩}.

\glsxtrnewglsdisp[⟨default-options⟩]{⟨prefix⟩}{⟨cs⟩}  glossaries-extra v1.49+
§5.7; 258

Defines the command \{⟨options⟩\}{⟨label⟩}{⟨text⟩} to behave like \glsdisp[⟨default-options⟩,⟨options⟩]{⟨prefix⟩⟨label⟩}{⟨text⟩}.

\glsxtrnewGLSlike[⟨default-options⟩]{⟨prefix⟩}{⟨\GLS-like cs⟩}{⟨\GLSpl-like cs⟩}  glossaries-extra v1.21+
§5.7; 257

Like \glsxtrnewgls but provides all caps commands.

\glsxtrnewglslike[⟨default-options⟩]{⟨prefix⟩}{⟨\gls-like cs⟩}{⟨\glspl-like cs⟩}{⟨\Gls-like cs⟩}{⟨\Glspl-like cs⟩}  glossaries-extra v1.21+
§5.7; 257

Like \glsxtrnewgls but provides plural and sentence case commands as well.
Command Summary

\glsxtrnewglslink\([\langle \text{default-options} \rangle \rangle \langle \text{prefix} \rangle \langle \text{cs} \rangle \) glossaries-extra v1.49+

§5.7; 258

Defines the command \( \langle \text{cs} \rangle [\langle \text{options} \rangle \rangle \langle \text{label} \rangle \langle \text{text} \rangle \) to behave like \glslink\([\langle \text{default-options} \rangle \rangle \langle \text{prefix} \rangle \langle \text{label} \rangle \langle \text{text} \rangle \).

\glsxtrnewnumber\([\langle \text{key=value list} \rangle \rangle \langle \text{entry-label} \rangle \langle \text{num} \rangle \) (requires \usepackage[\text{numbers}]{glossaries-extra})

§2.1; 11

Defines a new glossary entry with the given label, type set to numbers, the category set to number, the name set to \( \langle \text{num} \rangle \) and the sort set to \( \langle \text{entry-label} \rangle \). The optional argument is a comma-separated list of glossary entry keys, which can be used to override the defaults.

\glsxtrnewrgls\([\langle \text{default-options} \rangle \rangle \langle \text{prefix} \rangle \langle \text{cs} \rangle \) glossaries-extra v1.21+

§5.7; 258

Like \glsxtrnewgls but uses \rgls.

\glsxtrnewrGLSlike\([\langle \text{default-options} \rangle \rangle \langle \text{prefix} \rangle \langle \text{\textbackslash rGLS-like \text{cs}} \rangle \langle \text{\textbackslash rGLSpl-like \text{cs}} \rangle \) glossaries-extra v1.21+

§5.7; 259

Like \glsxtrnewrgls but provides all caps commands.

\glsxtrnewrglslike\([\langle \text{default-options} \rangle \rangle \langle \text{prefix} \rangle \langle \text{\textbackslash rGLS-like \text{cs}} \rangle \langle \text{\textbackslash rGLSpl-like \text{cs}} \rangle \langle \text{\textbackslash rGls-like \text{cs}} \rangle \langle \text{\textbackslash rGlspl-like \text{cs}} \rangle \) glossaries-extra v1.21+

§5.7; 258

Like \glsxtrnewrgls but provides plural and sentence case commands as well.

\glsxtrnewsymbol\([\langle \text{key=value list} \rangle \rangle \langle \text{entry-label} \rangle \langle \text{sym} \rangle \) (requires \usepackage[\text{symbols}]{glossaries-extra})

§2.1; 10

Defines a new glossary entry with the given label, type set to symbols, the category set to symbol, the name set to \( \langle \text{sym} \rangle \) and the sort set to \( \langle \text{entry-label} \rangle \). The optional argument is a comma-separated list of glossary entry keys, which can be used to override the defaults.
Command Summary

\glsxtrNoGlossaryWarning\glossary-type

Issues a warning with \GlossariesExtraWarning indicating that the given glossary is missing.

\GlsXtrNoGlsWarningAutoMake\glossary-label

Advisory message when `automake` has been used.

\GlsXtrNoGlsWarningBuildInfo

Build advice.

\GlsXtrNoGlsWarningCheckFile\file

Advisory message to check the file contents.

\GlsXtrNoGlsWarningEmptyMain

Produces the boilerplate text if the probably empty glossary is the main one.

\GlsXtrNoGlsWarningEmptyNotMain\glossary-label

Produces the boilerplate text if the probably empty glossary is not the main one.

\GlsXtrNoGlsWarningEmptyStart

Produces the boilerplate text if a glossary is probably empty.

\GlsXtrNoGlsWarningHead\glossary-label\file

848
Command Summary

Produces the header boilerplate text if a glossary file is missing.

\GlsXtrNoGlsWarningMisMatch

Advisory message on mis-matching \makenoidxglossaries.

\GlsXtrNoGlsWarningNoOut\{\file\}

Advisory if no output file was created.

\GlsXtrNoGlsWarningTail

Final paragraph of missing glossary boilerplate text.

\glsxtrnoidxgroups

Makes the group titling mechanism used with the “unsrt” family of commands use the same method as for \printnoidxglossary (ASCII only). This command can’t be used with \makeglossaries or with record.

\glsxtrnonprintablerules

Expands to non-printable character sort rules.

\glsxtrnopostpunc

When placed at the end of the description, this switches off the post-description punctuation (inserted automatically via options such as postdot) but doesn’t suppress the post-description hook. Does nothing outside of the glossary.
Expands to the name value for styles like `long-only-short-only-desc`.

\glxtronlydescsort \hspace{1em} glossaries-extra v1.17+

Expands to the name value for styles like `long-only-short-only-desc`.

\glxtronlyname \hspace{1em} glossaries-extra v1.25+

Expands to the name value for styles like `long-only-short-only`.

\glxtronlysuffix \hspace{1em} initial: \glxtrbrvplursuffix \hspace{1em} glossaries-extra v1.17+

The plural suffix used by the “only” abbreviation styles (such as `long-only-short-only`).

\glisxtrgkeylist \hspace{1em} glossaries-extra v1.49+

Expands to the original option list as it was supplied to \newabbreviation.

\glisxtrgshort \hspace{1em} glossaries-extra v1.17+

Expands to the original short form as it was supplied to \newabbreviation.

\glisxtrglong \hspace{1em} glossaries-extra v1.17+

Expands to the original long form as it was supplied to \newabbreviation.

\GLSxstrp\{\textit{field}\}\{-\langle\textit{entry-label}\rangle\} \hspace{1em} glossaries-extra v1.07+

As \glisxtrp but converts to uppercase (but not in the PDF bookmark).
Command Summary

\Glxtrp{⟨field⟩}{⟨entry-label⟩}
glossaries-extra v1.07+ §5.4; 232

As \glsxtrp but converts the first letter to uppercase (but not in the PDF bookmark).

\glsxtrp{⟨field⟩}{⟨entry-label⟩}
glossaries-extra v1.07+ §5.4; 231

For use in headings and captions (instead of the \gls-like or \glstext-like commands). This command is designed to expand to the field value if used in a PDF bookmark and can also expand to a more appropriate command if it ends up in the page header. Note that there’s no option argument. Options should be set beforehand using \glsxtrsetpopts, which is done automatically in the glossary with \glossxtrsetpopts.

\glsxtrpageref{⟨entry-label⟩}
glossaries-extra v1.11+ §8.1; 377

As \glsrefentry but uses \pageref instead of \ref. As with \glsrefentry, this will use \gls instead if the corresponding entry counter is disabled.

\glsxtrparen{⟨text⟩}
glossaries-extra v1.17+ 132

Used to encapsulate ⟨text⟩ in parentheses.

\Glsxtrpdfentryfmt{⟨entry-label⟩}{⟨text⟩}
glossaries-extra v1.49+ §5.12.2; 299

Does \MFUsentencecase{⟨text⟩}.

\Glsxtrpdfentryfmt{⟨entry-label⟩}{⟨text⟩}
glossaries-extra v1.42+ §5.12.2; 296

Just does ⟨text⟩.

\Glsxtrpl[⟨gls-options⟩][⟨dfn-options⟩]{⟨entry-label⟩}
§13; 579

As \glsxtrpl but applies sentence case.
Command Summary

\glsxtrpl{gls-options}{dfn-options}{entry-label}

As \glsxtr but shows the plural form.

\glsxtrpostabbrvfootnote{entry-label}{fmt-code} glossaries-extra v1.49+

Command used in the post-link hook for styles like short-postfootnote.

\glsxtrpostdescabbreviation
initial: empty

The default post-description hook for the abbreviation category.

\glsxtrpostdescacronym
initial: empty

The default post-description hook for the acronym category.

\glsxtrpostdesc\langle category\rangle
glossaries-extra

The post-description hook associated with the category identified by the label \langle category\rangle.

\glsxtrpostdescgeneral
initial: empty

The default post-description hook for the general category.

\glsxtrpostdescindex
\usepackage{index}{glossaries-extra}

initial: empty (requires \usepackage{index}{glossaries-extra})

The default post-description hook for the index category.

\glsxtrpostdescnumber
\usepackage{numbers}{glossaries-extra}

initial: empty (requires \usepackage{numbers}{glossaries-extra})

852
The default post-description hook for the **number** category.

\texttt{\textbackslash glsxtrpostdescription} \hspace{1cm} \S 8.6.2; 421

An additional hook used within \texttt{\textbackslash glspostdescription} that implements the category post-description hook.

\texttt{\textbackslash glsxtrpostdescsymbol} \hspace{1cm} initial: empty \hspace{1cm} (requires \texttt{usepackage[symbols]{glossaries-extra}}) \hspace{1cm} \S 2.1; 10

The default post-description hook for the **symbol** category.

\texttt{\textbackslash glsxtrpostdescterm} \hspace{1cm} initial: empty \hspace{1cm} \S 8.6.2; 422

The default post-description hook for the **term** category (which isn’t used by glossaries-extra).

\texttt{\textbackslash glsxtrpostfootnotelongformat\langle entry-label\rangle\langle fmt-cs\rangle} \hspace{1cm} glossaries-extra v1.49+ \hspace{1cm} 146

Used in the footnote text to format the long form for styles like **short-postfootnote**.

\texttt{\textbackslash GLSxtrposthyphenlong\langle entry-label\rangle\langle insert\rangle} \hspace{1cm} glossaries-extra v1.49+ \hspace{1cm} 153

As \texttt{\textbackslash glsxtrposthyphenlong} but all caps.

\texttt{\textbackslash glsxtrposthyphenlong\langle entry-label\rangle\langle insert\rangle} \hspace{1cm} glossaries-extra v1.17+ \hspace{1cm} 153

Used within the post-link hook to format the long form according to the **short-hyphen-postlong-hyphen** style on first use.

\texttt{\textbackslash GLSxtrposthyphenlongpl\langle entry-label\rangle\langle insert\rangle} \hspace{1cm} glossaries-extra v1.49+ \hspace{1cm} 154
Command Summary

As \glsxtrposthyphenlongpl but all caps.

\glsxtrposthyphenlongpl{⟨entry-label⟩}{⟨insert⟩}  
glossaries-extra v1.49+

As \glsxtrposthyphenlong but shows the plural.

\GLSxtrposthyphenshort{⟨entry-label⟩}{⟨insert⟩}  
glossaries-extra v1.49+

As \glsxtrposthyphenshort but all caps.

\glsxtrposthyphenshort{⟨entry-label⟩}{⟨insert⟩}  
glossaries-extra v1.17+

Used within the post-link hook to format the short form according to the long-hyphen-postshort-hyphen style on first use.

\GLSxtrposthyphenshortpl{⟨entry-label⟩}{⟨insert⟩}  
glossaries-extra v1.49+

As \glsxtrposthyphenshortpl but all caps.

\glsxtrposthyphenshortpl{⟨entry-label⟩}{⟨insert⟩}  
glossaries-extra v1.49+

As \glsxtrposthyphenshort but plural.

\GLSxtrposthyphensubsequent{⟨entry-label⟩}{⟨insert⟩}  
glossaries-extra v1.49+

As \glsxtrposthyphensubsequent but all caps.

\glsxtrposthyphensubsequent{⟨entry-label⟩}{⟨insert⟩}  
glossaries-extra v1.17+

Used within the post-link hook to format the insert according to the long-hyphen-postshort-hyphen style on subsequent use.

854
A post-link hook that does \glsxtrpostlink\langle category\rangle if that command has been defined, where the category label is obtained from the entry that has just been referenced with a \gls-like or \glstext-like command (using \glslabel). Does nothing if \glsxtrpostlink\langle category\rangle isn’t defined.

May be used within a post-link hook to display the description in parentheses on first use.

May be used within a post-link hook to display the symbol and description in parentheses on first use.

May be used within a post-link hook to display the symbol in parentheses on first use.

The post-link hook associated with the category identified by the label \langle category\rangle.

A post-link hook that’s used if a full stop is discarded in order to adjust the space factor (to denote the end of a sentence). If the category post-link hook exists, and will be applied and the full stop will be restored.
A post-link hook that checks if a following full stop needs to be discarded, in which case it does \glsxtrpostlinkendsentence, otherwise it does \glsxtrpostlink.

\glsxtrpostlinkSymbolDescSep glossaries-extra v1.49+  §5.5.4; 251

Used by \glsxtrpostlinkAddSymbolDescOnFirstUse to separate the symbol and description, if both are set.

\glslocalreset{⟨entry-label⟩} §5.10; 280

Hook performed by \glslocalreset. This hook is modified by \glsexableentrycount and \glsexableentryunitcount.

\glsxtrpostlocalunset{⟨entry-label⟩} §5.10; 280

Hook performed by \glsxtrpostlocalunset. This hook is modified by \glsexableentrycount and \glsexableentryunitcount.

\glsxtrpostlongdescription glossaries-extra v1.12+ §3.1; 32

Hook added to the end of the description field by the unstarred version of \longnewglossaryentry.

\glsxtrpostname{⟨category⟩} glossaries-extra v1.04+ §8.6.1; 420

The post-name hook associated with the category identified by the label ⟨category⟩.

\glsxtrpostnamehook{⟨entry-label⟩} §8.6.1; 420

A hook that’s performed within \glossentryname and \glossentrynameother after the entry name is displayed. This hook implements auto-indexing (see §12), then the general hook \glsextrapostnamehook and finally the \glsxtrpostname{⟨category⟩} hook.

856
Command Summary

\GlsXtrPostNewAbbreviation

A hook that's performed after the entry has been defined.

\glsxtrpostreset{⟨entry-label⟩}

Hook performed by \glsreset. This hook is modified by \glsenableentrycount and \glsenableentryunitcount.

\glsxtrpostunset{⟨entry-label⟩}

Hook performed by \glsunset. This hook is modified by \glsenableentrycount and \glsenableentryunitcount.

\glsxtrpostuserlongformat{⟨entry-label⟩}{⟨fmt-cs⟩} glossaries-extra v1.49+

Formats the long form in parentheses (with \glsxtruserparen) in styles like short-postlong-user.

\glsxtrpostusershortformat{⟨entry-label⟩}{⟨fmt-cs⟩} glossaries-extra v1.49+

Formats the short form in parentheses (with \glsxtruserparen) in styles like long-postshort-user.

\GlsXtrPrefixLabelFallbackLastfalse glossaries-extra-bib2gls v1.49+

Sets the \ifGlsXtrPrefixLabelFallbackLast conditional to false.

\GlsXtrPrefixLabelFallbackLasttrue glossaries-extra-bib2gls v1.49+

Sets the \ifGlsXtrPrefixLabelFallbackLast conditional to true.
\textbf{Command Summary}

\begin{center}
\begin{tabular}{|l|l|}
\hline
\texttt{\textbackslash glsxtrpreglossarystyle} & glossaries-extra v1.49+ \\
\hline
\texttt{\textbackslash glsxtrprelocation} & initial: \texttt{\textbackslash space} glossaries-extra-stylemods v1.21+ \\
\hline
\texttt{\textbackslash glsxtrprependlabelprefix}\{\langle label-prefix\rangle\} & glossaries-extra-bib2gls v1.37+ \\
\hline
\texttt{\textbackslash GlsXtrPreToDefaultGlsOpts}\{\langle options\rangle\} & glossaries-extra v1.49+ \\
\hline
\texttt{\textbackslash glsxtprovidelaccsupport}\{\langle category\rangle\}\{\langle field\rangle\} & (requires \texttt{accsupp}) \\
\hline
\texttt{\textbackslash GlsXtrProvideBibTeXFields} & glossaries-extra-bib2gls v1.29+ \\
\hline
\texttt{\textbackslash glsxtprovidecommand}\{\langle cs\rangle\}\{\langle n\rangle\}\{\langle default\rangle\}\{\langle definition\rangle\} & modifier: * \\
\hline
\end{tabular}
\end{center}

Hook performed by \texttt{\textbackslash setglossarystyle} to initialise default definitions of style commands.

Used before the location list in the predefined styles provided by glossaries-extra or modified by glossaries-extra-stylemods.

Prepends \langle label-prefix\rangle to the list of known labels.

Locally prepend \langle options\rangle to the default options for the \texttt{\textbackslash gls}-like and \texttt{\textbackslash glstext}-like commands.

Defines \texttt{\textbackslash glsxt\langle category\rangle\langle field\rangle accsupp} to \texttt{\textbackslash glsshortaccsupp}, if not already defined.

Provides the standard \LaTeX\ fields as glossary entry keys (using \texttt{\textbackslash glsaddstoragekey}).

Just uses \texttt{\textbackslash providecommand} within the \LaTeX\ document but is treated as \texttt{\textbackslash renewcommand} by bib2gls’s interpreter.
Command Summary

\glsxtrprovidestoragekey{⟨key⟩}{⟨default value⟩}{⟨no link cs⟩}  modifier: *  §3.2; 35

Like \glsaddstoragekey but does nothing if the key has already been defined.

\glsxtrrecentanchor  §11.5.6; 557

Defined by \glsxtrdisplaylocnameref to expand to the ⟨href⟩ argument. This corresponds to the value of \@currentHref when the record was created.

\GlsXtrRecordCount{⟨entry-label⟩}{⟨counter⟩}  §11.4; 523

Expands to the entry’s record count for the given counter (stored in the recordcount. ⟨counter⟩ field) or to 0 if not set.

\GlsXtrRecordCounter{⟨counter-name⟩}  §8.4.3.2; 407

(preamble only)

Activates recording for the given counter.

\glsxtrrecordtriggervalue{⟨entry-label⟩}  §11.4; 524

Expands to the trigger value used by \glsxtrifrecordtrigger.

\GlsXtrRecordWarning{⟨glossary-type⟩}  §5.5.2; 240

Incorrect use of \printglossary with non-hybrid record.

\glsxtrregularfont{⟨text⟩}  §5.5.2; 240

Used by \glsentryfmt to encapsulate regular entries. Also used by \glsxtrassignfield-font for regular entries.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\GlsXtrResetLocalBuffer</td>
<td>Glossary - extra v1.49+</td>
<td>§5.10.1; 285</td>
</tr>
</tbody>
</table>

If local unset for repeat entries has been enabled with \GlsXtrUnsetBufferEnableRepeatLocal, this will locally reset all entries that are in the buffer that hadn’t been marked as used before the function was enabled.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrresourcecount</td>
<td>Glossary - extra v1.12+</td>
<td>§11; 511</td>
</tr>
</tbody>
</table>

A count register that is incremented on each use of \GlsXtrLoadResources to provide a unique basename for each resource set.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrresourcefile{⟨options⟩}{⟨basename⟩}</td>
<td>Glossary - extra v1.11+</td>
<td>§11; 510</td>
</tr>
</tbody>
</table>

For use with \bib2gls, this both sets up the options for the resource set (which \bib2gls can detect from the aux file) and inputs the file \langle{basename}.glstex file created by \bib2gls.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrresourceinit</td>
<td>Glossary - extra v1.21+</td>
<td>§11; 512</td>
</tr>
</tbody>
</table>

May be defined to temporarily change command definitions before information is written to the aux file by the protected write used by \glsxtrresourcefile.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrrestoremarkhook</td>
<td></td>
<td>§5.3.3; 227</td>
</tr>
</tbody>
</table>

Counteracts \glsxtrmarkhook.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrrestorepostpunc</td>
<td>Glossary - extra v1.23+</td>
<td>§8.6.2; 423</td>
</tr>
</tbody>
</table>

Used in the description to counteract the use of \glsxtrnopostpunc. Does nothing outside of the glossary.
Command Summary

Style-sensitive abbreviation command designed to counteract any font change applied by the style.

\texttt{\glsxtrRevertMarks}

Restores \texttt{\markright}, \texttt{\markboth} and \texttt{\@starttoc} to their previous definitions.

\texttt{\glsxtrRevertTocMarks}

Restores \texttt{\@starttoc} to its previous definition.

\texttt{\glsxtrsaveinsert}{⟨entry-label⟩}{⟨insert⟩}

Implemented at the start of all the \texttt{\glstext}-like commands (except the inline full form commands like \texttt{\glsxtrfull}) to save the \texttt{\glsinsert} placeholder. By default, this sets \texttt{\glsinsert} to empty.

\texttt{\glsxtrscfont}{⟨text⟩}

Maintained for backwards-compatibility used to typeset ⟨text⟩ in small capitals (\texttt{\textsc}) for the “sc” abbreviation styles.

\texttt{\glsxtronlydescname}

Expands to the name value for styles like \texttt{long-only-short-sc-only-desc}.

\texttt{\glsxtronlydescsort}

Expands to the sort value for styles like \texttt{long-only-short-sc-only-desc}.

\texttt{\glsxtronlyname}
Command Summary

Expands to the name value for styles like `long-only-short-sc-only`.

\glsxtronlyrevert\{⟨text⟩\}  
\texttt{glossaries-extra v1.49+}

The definition of \glsxtrrevert used by styles like `long-only-short-sc-only`. Uses \glsxtrcrevert.

\glsxtronlysuffix  \texttt{initial: \glsxtrcsuffix glossaries-extra v1.48+}

The plural suffix used by the “sc-only” abbreviation styles (such as `long-only-short-sc-only`).

\glsxtrcrevert\{⟨text⟩\}  
\texttt{glossaries-extra v1.49+}

The definition of \glsxtrrevert used by the small caps (“sc”) abbreviation styles. Uses \glistextup.

\glsxtrcsuffix  
\texttt{glossaries-extra v1.48+}

The plural suffix used by the small caps (“sc”) abbreviation styles. This switches off the small caps font to prevent the suffix from also appearing in small caps.

\glsxtrcuserrevert\{⟨text⟩\}  
\texttt{glossaries-extra v1.49+}

The definition of \glsxtrrevert used by styles like `long-postshort-sc-user`. Uses \glsxtrcrevert.

\glsxtrcusersuffix  
\texttt{glossaries-extra v1.48+}

The plural suffix used by styles like `long-postshort-sc-user`.

\glsxtrseealsolabels\{⟨entry-label⟩\}  
\texttt{glossaries-extra v1.16+; §5.9.2; 277}  
\texttt{862}
Command Summary

Expands to the value of the seealso field for the entry identified by \langle entry-label \rangle. If the field isn’t set, this will expand to nothing. If the entry isn’t defined, this will expand to \relax.

\glsxtrseelists\{csv-list\} \hspace{1cm} glossaries-extra v1.16+ §5.13; 299

Fully expands \langle csv-list \rangle and passes it to \glsseelists.

\glsxtrseelists\{entry-label\} \hspace{1cm} glossaries-extra v1.49+ §5.9.2; 276

If the entry given by \langle entry-label \rangle has the see, seealso or alias fields set, this will display the cross reference according to \glsxtruseeseeformat (for see and alias) or \glsxtrusesealsoformat (for seealso). If any of these fields are set, the list is encapsulated with \glsxtrseelistsencap.

\glsxtrseelistsdelim \hspace{1cm} glossaries-extra v1.49+ §5.9.2; 277

Used by \glsxtrseelists to as separator between sub-lists.

\glsxtrseelistsencap\{content\} \hspace{1cm} glossaries-extra v1.49+ §5.9.2; 276

Used by \glsxtrseelists to encapsulate the lists.

\glsxtrsetactualanchor\{counter\} \hspace{1cm} glossaries-extra-bib2gls v1.49+ §11.5.6; 557

Hook used by \glsxtrdisplaylocnameref to override the default definition of \glsxtractualanchor.

\GlsXtrSetActualChar\{character\} \hspace{1cm} (preamble only) §12; 577

Sets the “actual character” for the auto-indexing feature.
# Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrsetaliasnoindex</td>
<td>Hook used to switch off indexing for aliases.</td>
<td>glossaries-extra v1.12+</td>
</tr>
<tr>
<td>\GlsXtrSetAltModifier{⟨token⟩}{⟨options⟩}</td>
<td>Sets ⟨token⟩ as a modifier for the \gls-like and \glstext-like commands that will automatically implement the given options.</td>
<td>§5; 184</td>
</tr>
<tr>
<td>\glsxtrsetbibglsaux{⟨basename⟩}</td>
<td>As the \bibglsaux option.</td>
<td>glossaries-extra v1.49+ (requires bib2gls v3.0+)</td>
</tr>
<tr>
<td>\glsxtrsetcategory{⟨entry-labels⟩}{⟨category-label⟩}</td>
<td>Globally sets the category field to the fully expanded ⟨category-label⟩ for each entry listed in ⟨entry-labels⟩.</td>
<td>§10; 492</td>
</tr>
<tr>
<td>\glsxtrsetcategoryforall{⟨glossary-labels⟩}{⟨category-label⟩}</td>
<td>Globally sets the category field to the fully expanded ⟨category-label⟩ for each entry belonging to the glossaries listed in ⟨glossary-labels⟩.</td>
<td>§10; 492</td>
</tr>
<tr>
<td>\glsxtrsetcomplexstyle{⟨entry-label⟩}{⟨n⟩}</td>
<td>Indicates that the entry given by ⟨entry-label⟩ uses a complex abbreviation style. The second argument ⟨n⟩ should be numeric, which indicates why it doesn’t work with the variations of \glsfirst: 1 (all caps doesn’t work), 2 (all caps and insert doesn’t work), 3 (insert doesn’t work).</td>
<td>glossaries-extra v1.49+ §4.5.3.1; 166</td>
</tr>
<tr>
<td>\GlsXtrSetDefaultGlsOpts{⟨options⟩}</td>
<td></td>
<td>§5.1.1; 187</td>
</tr>
</tbody>
</table>
Command Summary

Locally set the default options for the \gls-like and \glstext-like commands.

\GlsXtrSetDefaultNumberFormat\{\langle encap \rangle \}\hspace{1em} glossaries-extra v1.19+\hspace{1em}§5.1.1; 188

Sets the default format to \langle encap \rangle (without the leading backslash).

\GlsXtrSetEncapChar\{\langle character \rangle \}\hspace{1em} (preamble only)\hspace{1em}§12; 577

Sets the “encap character” for the auto-indexing feature.

\GlsXtrSetEscChar\{\langle character \rangle \}\hspace{1em} (preamble only)\hspace{1em}§12; 577

Sets the “escape character” for the auto-indexing feature.

\GlsXtrSetField\{\langle entry-label \rangle \}\{\langle field-label \rangle \}\{\langle value \rangle \}\hspace{1em} glossaries-extra v1.12+\hspace{1em}§3.5; 39

Assigns \langle value \rangle to the field identified by its internal label \langle field-label \rangle for the entry identified by \langle entry-label \rangle. An error (or warning with undefaction=warn) occurs if the entry hasn’t been defined.

\glsxtrsetfieldifexists\{\langle entry-label \rangle \}\{\langle field-label \rangle \}\{\langle code \rangle \}\hspace{1em} glossaries-extra v1.12+\hspace{1em}§3.5; 39

Used by commands like \GlsXtrSetField to check if the entry exists before assigning a value to the field. The \langle code \rangle part is the assignment code, which is only done if the required condition is met. This can be redefined if the condition needs to be altered.

\glsxtrsetgrouptitle\{\langle group-label \rangle \}\{\langle group-title \rangle \}\hspace{1em} glossaries-extra v1.14+\hspace{1em}§8.6.4; 426

Globally assigns the given title \langle group-title \rangle to the group identified by \langle group-label \rangle.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glsxtrsetglossarylabel{⟨label⟩}</code></td>
<td>Sets the label to add (using <code>\label{⟨label⟩}</code>) after the glossary section heading.</td>
<td>§8.3; 378</td>
</tr>
<tr>
<td><code>\GlsXtrSetLevelChar{⟨character⟩}</code></td>
<td>Sets the “level character” for the auto-indexing feature.</td>
<td>§12; 577</td>
</tr>
<tr>
<td><code>\glsxtrsetlongfirstuse{⟨entry-label⟩}</code></td>
<td>Implemented by the <code>\glsxtrlong</code> set of commands to assign <code>\glsxtrifwasfirstuse</code></td>
<td>§4.3; 52</td>
</tr>
<tr>
<td><code>\GlsXtrSetPlusModifier{⟨options⟩}</code></td>
<td>Overrides the options that should be implemented by the plus (+) modifier for <code>\gls</code>-like and <code>\glstext</code>-like commands.</td>
<td>§5; 184</td>
</tr>
<tr>
<td><code>\glsxtrsetpopts{⟨options⟩}</code></td>
<td>Sets the options that <code>\glsxtrp</code> (and case-change variants) pass to the relevant <code>\glstext</code>-like command.</td>
<td>§5.4; 231</td>
</tr>
<tr>
<td><code>\glsxtrsetpunctuationmarks{⟨token list⟩}</code></td>
<td>Sets the punctuation list used by <code>\glsxtrifnextpunc</code>. The ⟨token list⟩ must be a non-delimited list of single tokens that represent each punctuation character. Note that the element of the list must be a single token, which means a single-byte character for pdfLaTeX (for example, ASCII). Multi-byte characters (UTF-8) will required a native Unicode engine (XeLaTeX or LuaLaTeX).</td>
<td>§5.5.4; 249</td>
</tr>
<tr>
<td><code>\GlsXtrSetRecordCountAttribute{⟨category-list⟩}{⟨value⟩}</code></td>
<td></td>
<td>§11.4; 524</td>
</tr>
</tbody>
</table>
Command Summary

Sets the `recordcount` attribute to `<value>` for each of the listed categories.

\texttt{\GlsXtrSetStarModifier\{⟨options⟩\}} \hspace{1cm} glossaries-extra v1.49+

Overrides the options that should be implemented by the star (*) modifier for \texttt{gls}-like and \texttt{glstext}-like commands.

\texttt{\glsxtrsetupfulldefs} \hspace{1cm} §4.3; 54

Hook used by \texttt{\glsxtrfull} (and case-changing and plural variations).

\texttt{\glsxtrSetWidest\{⟨type⟩\}\{⟨level⟩\}} \hspace{1cm} glossaries-extra-bib2gls v1.37+

Written to the \texttt{glstex} by the `set-widest` option.

\texttt{\glsxtrSetWidestFallback\{⟨type⟩\}\{⟨level⟩\}} \hspace{1cm} glossaries-extra-bib2gls v1.37+

Written to the \texttt{glstex} by the `set-widest` option if bib2gls can’t determine the widest name.

\texttt{\GLSxtrshort\[⟨options⟩\]\{⟨entry-label⟩\}\{⟨insert⟩\}} \hspace{1cm} modifiers: * + ⟨alt-mod⟩ §4.3; 52

As \texttt{\glsxtrshort} but converts the link text to all caps.

\texttt{\GLSxtrshort\[⟨options⟩\]\{⟨entry-label⟩\}\{⟨insert⟩\}} \hspace{1cm} modifiers: * + ⟨alt-mod⟩ §4.3; 51

As \texttt{\glsxtrshort} but converts the first character of the link text to uppercase (for the start of a sentence) using \texttt{\makefirstuc}. 

\texttt{\glsxtrshort\[⟨options⟩\]\{⟨entry-label⟩\}\{⟨insert⟩\}} \hspace{1cm} modifiers: * + ⟨alt-mod⟩ §4.3; 51
References the entry identified by \langle\textit{entry-label}\rangle. The text produced is obtained from the \texttt{short} value, formatted according to the abbreviation style associated with the entry’s category. The \langle\textit{insert}\rangle argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see \texttt{\glslink} options.

\begin{verbatim}
\glsxtrshortdescname\text{ \texttt{glossaries-extra v1.17+}}
\end{verbatim}

Expands to the name value for styles like \texttt{short-nolong-desc}.

\begin{verbatim}
\GLSxtrshortformat\text{\langle\textit{entry-label}\rangle\langle\textit{insert}\rangle\langle\texttt{fmt-cs}\rangle \text{ \texttt{glossaries-extra v1.49+}}}
\end{verbatim}

As \texttt{\glsxtrshortformat} but all caps.

\begin{verbatim}
\Glsxtrshortformat\text{\langle\textit{entry-label}\rangle\langle\textit{insert}\rangle\langle\texttt{fmt-cs}\rangle \text{ \texttt{glossaries-extra v1.49+}}}
\end{verbatim}

As \texttt{\glsxtrshortformat} but sentence case.

\begin{verbatim}
\glsxtrshortformat\text{\langle\textit{entry-label}\rangle\langle\textit{insert}\rangle\langle\texttt{fmt-cs}\rangle \text{ \texttt{glossaries-extra v1.49+}}}
\end{verbatim}

Encapsulates the \texttt{short} field for the given entry with \langle\texttt{fmt-cs}\rangle. The \langle\textit{insert}\rangle argument is the insertion material supplied in the final optional argument of the \texttt{\gls}-like or \texttt{\glstext}-like commands. The \texttt{\ifglsxtrinsertinside}, inner formatting, and accessibility settings are supported.

\begin{verbatim}
\GLSxtrshortformatgrp\text{\langle\textit{entry-label}\rangle\langle\textit{insert}\rangle\langle\texttt{fmt-cs}\rangle \text{ \texttt{glossaries-extra v1.49+}}}
\end{verbatim}

As \texttt{\glsxtrshortformatgrp} but all caps.

\begin{verbatim}
\Glsxtrshortformatgrp\text{\langle\textit{entry-label}\rangle\langle\textit{insert}\rangle\langle\texttt{fmt-cs}\rangle \text{ \texttt{glossaries-extra v1.49+}}}
\end{verbatim}

As \texttt{\glsxtrshortformatgrp} but sentence case.
**Command Summary**

\glsxtrshortformatgrp{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}  
\texttt{glossaries-extra v1.49+}

As \glsxtrshortformat but adds grouping around ⟨insert⟩ (with the inner formatting inside the group).

\glsxtrshorthyphen{⟨short⟩}{⟨entry-label⟩}{⟨insert⟩}  
\texttt{glossaries-extra v1.17+}

Formats the short form according to the short–hyphen–postlong–hyphen style.

\GLSxtrshorthyphenlong{⟨entry-label⟩}{⟨short⟩}{⟨long⟩}{⟨insert⟩}  
\texttt{glossaries-extra v1.49+}

As \glsxtrshorthyphenlong but ⟨insert⟩ is converted to all caps.

\glsxtrshorthyphenlong{⟨entry-label⟩}{⟨short⟩}{⟨long⟩}{⟨insert⟩}  
\texttt{glossaries-extra v1.17+}

Formats the short and long form according to the short–hyphen–long–hyphen style.

\glsxtrshorthyphenlongsort  
\texttt{glossaries-extra v1.49+}

Expands to the sort value for the short–hyphen–long–hyphen styles.

\glsxtrshortlongdescname  
\texttt{glossaries-extra v1.17+}

Expands to the name value for short–long–desc styles.

\glsxtrshortlongdescsort  
\texttt{glossaries-extra v1.17+}

Expands to the sort value for short–long–desc styles.
Command Summary

\GLSxtrshortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrshortlongformat but all caps.

\Glsxtrshortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrshortlongformat but sentence case.

\glsxtrshortlongformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+

Formats the short form with \glsxtrshortformat and the long form in parentheses with \glsxtrlongformat.

\glsxtrshortlongname
glossaries-extra v1.25+

Expands to the name value for short-long styles.

\GLSxtrshortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrshortlongplformat but all caps.

\Glsxtrshortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+

As \glsxtrshortlongplformat but sentence case.

\glsxtrshortlongplformat{⟨entry-label⟩}{⟨insert⟩}{⟨long-fmt-cs⟩}{⟨short-fmt-cs⟩}
glossaries-extra v1.49+
## Command Summary

As `\glsxtrshortlongformat` but for the plurals.

\begin{itemize}
  \item \texttt{\glsxtrshortlonguserdescname} \gls{glossaries-extra v1.25+}
\end{itemize}

Expands to the value for the \texttt{name} key for styles like \texttt{short-long-user-desc}.

\begin{itemize}
  \item \texttt{\glsxtrshortnolongname} \gls{glossaries-extra v1.25+}
\end{itemize}

Expands to the name value for \texttt{short-nolong} styles.

\begin{itemize}
  \item \texttt{\Glsxtrshortpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} \textbf{modifiers: } * + ⟨alt-mod⟩ \gls{§4.3; 52}
\end{itemize}

As `\glsxtrshort` but converts the link text to all caps.

\begin{itemize}
  \item \texttt{\Glsxtrshortpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} \textbf{modifiers: } * + ⟨alt-mod⟩ \gls{§4.3; 52}
\end{itemize}

As `\glsxtrshortpl` but converts the first character of the link text to uppercase (for the start of a sentence) using `\makefirstuc`.

\begin{itemize}
  \item \texttt{\Glsxtrshortpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} \textbf{modifiers: } * + ⟨alt-mod⟩ \gls{§4.3; 52}
\end{itemize}

References the entry identified by ⟨entry-label⟩. The text produced is obtained from the \texttt{shortplural} value, formatted according to the abbreviation style associated with the entry’s category. The ⟨insert⟩ argument will be inserted at the end of the link text. This command does not alter or depend on the first use flag. For the first optional argument, see `\glslink` options.

\begin{itemize}
  \item \texttt{\Glsxtrshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}} \gls{glossaries-extra v1.49+}
\end{itemize}

As `\glsxtrshortplformat` but all caps.

\begin{itemize}
  \item \texttt{\Glsxtrshortplformat{⟨entry-label⟩}{⟨insert⟩}{⟨fmt-cs⟩}} \gls{178}
\end{itemize}
As \texttt{\glsxtrshortplformat}\{⟨entry-label⟩}⟨insert⟩⟨fmt-cs⟩\} but for the \texttt{shortplural} field.

As \texttt{\glsxtrshortplformatgrp}\{⟨entry-label⟩}⟨insert⟩⟨fmt-cs⟩\} but all caps.

As \texttt{\glsxtrshortplformatgrp}\{⟨entry-label⟩}⟨insert⟩⟨fmt-cs⟩\} but sentence case.

As \texttt{\glsxtrshortplformat} but adds grouping around ⟨insert⟩ (with the inner formatting inside the group).

Used in inner mode for debugging, this defaults to \texttt{\glsxshowtargetinner} but is changed by the \texttt{showtargets} options.

\texttt{\glsxshowtargetouter}\{⟨target-name⟩\}
**Command Summary**

Used in outer mode for debugging, this defaults to `\glsshowtargetouter` but is changed by the `showtargets` options.

\begin{itemize}
  \item \texttt{\glsxtrshowtargetsymbolleft}
  \hspace{1cm} glossaries-extra v1.48+  \\
  The left marker debugging symbol ($\lhd$).
  \item \texttt{\glsxtrshowtargetsymbolright}
  \hspace{1cm} glossaries-extra v1.48+  \\
  The right marker debugging symbol ($\triangleright$).
  \item \texttt{\glsxtrsmfont\{text\}}
  Maintained for backwards-compatibility used to typeset \langle text \rangle in a smaller font (\textsmaller) for the “sm” abbreviation styles.
  \item \texttt{\glsxtrrevert\{text\}}
  \hspace{1cm} glossaries-extra v1.49+  \\
  The definition of \texttt{\glsxtrrevert} used by the smaller ("sm") abbreviation styles. Uses \textlarger.
  \item \texttt{\glsxtrmsuffix}
  \hspace{1cm} initial: \texttt{\glsxtrabbrvpluralsuffix}  \\
  The plural suffix used by the smaller ("sm") abbreviation styles (such as \texttt{short-sm-long}).
  \item \texttt{\glsxtrspacerules}
  \hspace{1cm} glossaries-extra-bib2gls v1.27+  \\
  Expands to space character sort rules.
  \item \texttt{\GlsXtrStandaloneEntryHeadName\{entry-label\}}
  \hspace{1cm} glossaries-extra v1.49+  \\
  \end{itemize}
Command Summary

Used by \glsxtrglossentry for the header and toc.

\GlsXtrStandaloneEntryHeadOther{⟨entry-label⟩}{⟨field-label⟩}
glossaries-extra v1.49+

§8.5; 418

Used by \glsxtrglossentryother for the header and toc.

\GlsXtrStandaloneEntryName{⟨entry-label⟩}
glossaries-extra v1.37+

§8.5; 415

Used to display the standalone entry name and create the associated hypertarget, if supported.

\GlsXtrStandaloneEntryOther{⟨entry-label⟩}{⟨field-label⟩} glossaries-extra v1.37+

§8.5; 415

As \GlsXtrStandaloneEntryName but where the text is obtained from the given field instead of name.

\GlsXtrStandaloneEntryPdfName{⟨entry-label⟩}
glossaries-extra v1.49+

§8.5; 417

Used by \glsxtrglossentry for the PDF bookmark.

\GlsXtrStandaloneEntryPdfOther{⟨entry-label⟩}{⟨field-label⟩} glossaries-extra v1.49+

§8.5; 417

Used by \glsxtrglossentryother for the PDF bookmark.

\GlsXtrStandaloneGlossaryType
glossaries-extra v1.21+

§8.5; 414

Expands to the glossary type for standalone entries.

\GlsXtrStandaloneSubEntryItem{⟨entry-label⟩}
glossaries-extra v1.21+

§8.5; 414
## Command Summary

Used to display standalone entries that have the `parent` field set.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrstarflywarn</td>
<td>Issues a warning with \GlossariesExtraWarning indicating that the experimental starred version of \GlsXtrEnableOnTheFly has been used.</td>
<td></td>
</tr>
<tr>
<td>\GlsXtrStartUnsetBuffering</td>
<td>Enables unset buffering. The starred version doesn’t check for duplicates.</td>
<td>§5.10.1; 284</td>
</tr>
<tr>
<td>\GlsXtrStopUnsetBuffering</td>
<td>Stops buffering. The starred version performs a global unset.</td>
<td>§5.10.1; 285</td>
</tr>
<tr>
<td>\glsxtrSubScriptDigitrules</td>
<td>Expands to the 0–9 subscript digit character sort rules.</td>
<td>545</td>
</tr>
<tr>
<td>\GLSxtrsubsequentfmt{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Used by \glsxtrgenabbrvfmt to display the all caps subsequent singular form (defined by the abbreviation style).</td>
<td>173</td>
</tr>
<tr>
<td>\glsxtrsubsequentfmt{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Used by \glsxtrgenabbrvfmt to display the sentence case subsequent singular form (defined by the abbreviation style).</td>
<td>173</td>
</tr>
</tbody>
</table>
Command Summary

Used by \glsxtrgenabbrvfmt to display the subsequent singular form (defined by the abbreviation style).

\GLSxtrsubsequentplfmt\{⟨entry-label⟩\}\{⟨insert⟩\}   glossaries-extra v1.49+

Used by \glsxtrgenabbrvfmt to display the all caps subsequent plural form (defined by the abbreviation style).

\Glsxtrsubsequentplfmt\{⟨entry-label⟩\}\{⟨insert⟩\}   glossaries-extra v1.17+

Used by \glsxtrgenabbrvfmt to display the sentence case subsequent plural form (defined by the abbreviation style).

\glsxtrsubsequentplfmt\{⟨entry-label⟩\}\{⟨insert⟩\}   glossaries-extra v1.17+

Used by \glsxtrgenabbrvfmt to display the subsequent plural form (defined by the abbreviation style).

\glsxtrSuperScriptDigitrules   glossaries-extra-bib2gls v1.27+

Expands to the 0–9 superscript digit character sort rules.

\glsxtrsupphypernumber\{⟨location⟩\}   glossaries-extra v1.14+

§5.1.2; 195

Used to hyperlink to a location in an external document if the externallocation attribute has been set. This will define \glsxtrsupplocationurl to the location provided by the attribute or to empty if the attribute isn’t set.

\glsxtrsupplocationurl   glossaries-extra v1.14+

Defined by \glsxtrsupphypernumber to the external location or empty if not provided.


Command Summary

\texttt{\textbackslash glsxtrtagfont\{\langle text\rangle\}}

§4.4; 58

Used by the tagging command defined with \texttt{\textbackslash GlsXtrEnableInitialTagging}.

\texttt{\textbackslash glsxtrtaggedlist\{\langle singular tag\rangle\}{\langle plural tag\rangle\}{\langle label prefix\rangle\}{\langle csv-list\rangle}}

glossaries-extra v1.49+

§5.13; 300

Similar to \texttt{\textbackslash glsseelist}, this will start the list with \texttt{\langle singular tag\rangle} if the list only contains one element and \texttt{\langle plural tag\rangle} if the list contains more than one element. Each element is prefixed with \texttt{\langle label prefix\rangle}. The tag is separated from the start of the list with \texttt{\textbackslash glsxtrtaggedlist-sep}. The actual list separators as as for \texttt{\textbackslash glsseelist}. The \texttt{\langle csv-list\rangle} is expanded before being iterated over. Does nothing if \texttt{\langle csv-list\rangle} is empty.

\texttt{\textbackslash glsxtrtaggedlistsep}

\textit{initial: \textbackslash space}  
geossaries-extra v1.49+

§5.13; 300

Separator used by \texttt{\textbackslash glsxtrtaggedlist} between the tag and the list.

\texttt{\textbackslash GlsXtrTheLinkCounter\{\langle entry-label\rangle\}}

glossaries-extra v1.26+

§6.2; 318

Expands to the value of the link counter associated with the given entry or 0 if it hasn’t been defined.

\texttt{\textbackslash glsxtrtitlednameref\{\langle format\rangle\}{\langle location\rangle\}{\langle title\rangle\}{\langle file\rangle\}}

glossaries-extra-bib2gls v1.49+

§11.5.6; 559

Used by \texttt{\textbackslash glsxtrdisplaylocnameref} to display locations that have a title and are not associated with the page counter and don’t have an associated \texttt{\textbackslash glsxtr\{counter\}locfmt} command. The anchor is obtained from \texttt{\textbackslash glsxtrrecentanchor}.

\texttt{\textbackslash GLSxtrtitlefirst\{\langle entry-label\rangle\}}

glossaries-extra v1.42+

§5.3.3; 226

Used to display the all caps entry’s \texttt{first} field in the section title and table of contents.
\glsxtrtitlefirst\{⟨entry-label⟩\}

Used to display the sentence case entry’s first field in the section title and table of contents.

\glsxtrtitlefirst\{⟨entry-label⟩\}

Used to display the entry’s first field in the section title and table of contents.

\GLSxtrtitlefirstplural\{⟨entry-label⟩\}

Used to display the all caps entry’s firstplural field in the section title and table of contents.

\GLSxtrtitlefirstplural\{⟨entry-label⟩\}

Used to display the sentence case entry’s firstplural field in the section title and table of contents.

\glxtrtitlefirstplural\{⟨entry-label⟩\}

Used to display the entry’s firstplural field in the section title and table of contents.

\GLSxtrtitlefull\{⟨entry-label⟩\}

Used to display the entry’s all caps full form in the section title and table of contents.

\GLSxtrtitlefull\{⟨entry-label⟩\}

Used to display the entry’s sentence case full form in the section title and table of contents.
Command Summary

\glsxtrtitlefull{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.02+} \hspace{1cm} §5.3.3; 221

Used to display the entry’s full form in the section title and table of contents.

\GLSxtrtitlefullpl{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.42+} \hspace{1cm} §5.3.3; 223

Used to display the entry’s all caps full plural form in the section title and table of contents.

\Glsxtrtitlefullpl{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.02+} \hspace{1cm} §5.3.3; 222

Used to display the entry’s sentence case full plural form in the section title and table of contents.

\glsxtrtitlefullpl{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.02+} \hspace{1cm} §5.3.3; 222

Used to display the entry’s full plural form in the section title and table of contents.

\GLSxtrtitlelong{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.42+} \hspace{1cm} §5.3.3; 220

The normal behaviour of \GLSfmtlong.

\Glsxtrtitlelong{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.02+} \hspace{1cm} §5.3.3; 220

The normal behaviour of \Glsfmtlong.

\glsxtrtitlelong{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.02+} \hspace{1cm} §5.3.3; 220

The normal behaviour of \glsfmtlong.

\GLSxtrtitlelongpl{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.42+} \hspace{1cm} §5.3.3; 221

The normal behaviour of \GLSfmtlong. 

879
The normal behaviour of \GLSfmtlongpl.

\GLSxtrtitlelongpl\{\langle entry-label\rangle\} \hspace{1cm} {\text{glossaries-extra v1.02+}} \hspace{1cm} §5.3.3; 221

The normal behaviour of \Glsfmtlongpl.

\glsxtrtitlelongpl\{\langle entry-label\rangle\} \hspace{1cm} {\text{glossaries-extra v1.02+}} \hspace{1cm} §5.3.3; 221

The normal behaviour of \glssfmtlongpl.

\GLSXtrtitlename\{\langle entry-label\rangle\} \hspace{1cm} {\text{glossaries-extra v1.42+}} \hspace{1cm} §5.3.3; 223

Used to display the all caps entry’s name in the section title and table of contents.

\GLSXtrtitlename\{\langle entry-label\rangle\} \hspace{1cm} {\text{glossaries-extra v1.21+}} \hspace{1cm} §5.3.3; 223

Used to display the sentence case entry’s name in the section title and table of contents.

\glsxtrtitlename\{\langle entry-label\rangle\} \hspace{1cm} {\text{glossaries-extra v1.21+}} \hspace{1cm} §5.3.3; 223

Used to display the entry’s name in the section title and table of contents.

\glsxtrtitleopts \hspace{1cm} {\text{glossaries-extra v1.49+}} \hspace{1cm} §5.3.2; 206

Expands to the options that commands like \glssfmtshort should use in the title or caption within the document text.

\glsxtrtitleorpdforheading\{\langle title\rangle\}\{\langle PDF bookmarks\rangle\}\{\langle heading\rangle\} \hspace{1cm} {\text{glossaries-extra v1.21+}} \hspace{1cm} §5.3.3; 215

880
Command Summary

Does the applicable argument depending on whether the command occurs within a title/caption or PDF bookmark or heading.

\GLSxtrtitleplural\{⟨entry-label⟩\} glossaries-extra v1.42+ §5.3.3; 225

Used to display the all caps entry’s plural field in the section title and table of contents.

\Glsxtrtitleplural\{⟨entry-label⟩\} §5.3.3; 225

Used to display the sentence case entry’s plural field in the section title and table of contents.

\glsxtrtitleplural\{⟨entry-label⟩\} §5.3.3; 224

Used to display the entry’s plural field in the section title and table of contents.

\GLSxtrtitleshort\{⟨entry-label⟩\} glossaries-extra v1.42+ §5.3.3; 218

The normal behaviour of \GLSfmtshort.

\Glsxtrtitleshort\{⟨entry-label⟩\} §5.3.3; 217

The normal behaviour of \Glsfmtshort.

\glsxtrtitleshort\{⟨entry-label⟩\} §5.3.3; 217

The normal behaviour of \glssfmtshort.

\GLSxtrtitleshortpl\{⟨entry-label⟩\} glossaries-extra v1.42+ §5.3.3; 220
Command Summary

The normal behaviour of \GLSfmtshortpl.

\Glsxtrtitleshortpl{(entry-label)}

The normal behaviour of \Glsfmtshortpl.

\glsxtrtitleshortpl{(entry-label)}
glossaries-extra v1.03+

The normal behaviour of \glsfmtshortpl.

\GLSxtrtitletext{(entry-label)}
glossaries-extra v1.42+

Used to display the all caps entry’s text field in the section title and table of contents.

\Glsxtrtitletext{(entry-label)}

Used to display the sentence case entry’s text field in the section title and table of contents.

\glsxtrtitletext{(entry-label)}

Used to display the entry’s text field in the section title and table of contents.

\GlsXtrTotalRecordCount{(entry-label)}
glossaries-extra v1.21+

Expands to the entry’s total record count (stored in the recordcount field) or to 0 if not set.

\glsxtrtreechildpredesc initial: \glstreechildpredesc
glossaries-extra-stylemods v1.46+

Inserted before the child descriptions for the tree styles.
\texttt{\textbackslash glsxtrtreepredesc} \textit{initial: \textbackslash glstreepredesc} glossaries-extra-stylemods v1.46+

Inserted before the top-level descriptions for the tree styles.

\texttt{\textbackslash glsxtrundefaction\{\textit{message}\}\{\textit{additional help}\}} glossaries-extra v1.08+

§2.4; 16

Will either produce an error or a warning, depending on the \texttt{undefaction} setting. In the document environment this will also generate the unknown marker (??).

\texttt{\textbackslash glsxtrundeftag} \textit{initial: ??} glossaries-extra v1.08+

§2.4; 15

Expands to the unknown marker (??).

\texttt{\textbackslash GlsXtrUnknownDialectWarning\{\textit{locale}\}\{\textit{root language}\}} glossaries-extra v1.32+

§5.12.1; 294

Issues a warning with \texttt{\GlossariesExtraWarning} indicating that a valid dialect label can't be determined for the given locale and root language.

\texttt{\textbackslash GlsXtrUnsetBufferDisableRepeatLocal} glossaries-extra v1.49+

§5.10.1; 286

Disables \texttt{GlsXtrUnsetBufferEnableRepeatLocal}.

\texttt{\textbackslash GlsXtrUnsetBufferEnableRepeatLocal} glossaries-extra v1.49+

§5.10.1; 285

Allows repeat entries within the buffering code to be locally unset before the link text.

\texttt{\textbackslash glsxtrunsrtdo\{\textit{entry-label}\}} glossaries-extra v1.12+

§8.4.3; 402

Used by the "unsrt" family of commands, this displays the glossary entry according to the current glossary style (taking the hierarchical level into account, which may have been adjusted by \texttt{leveloffset} or \texttt{flatten}).

883
\glsxtrunusedformat{\langle location \rangle}

The format used by \glsxtraddallcrossrefs.

\glsxtrUpAlpha \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright alpha.

\glsxtrUpBeta \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright beta.

\glsxtrUpChi \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright chi.

\glsxtrUpDelta \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright delta.

\glsxtrUpDigamma \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright digamma.

\glsxtrUpEpsilon \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright epsilon.

\glsxtrUpEta \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright eta.
Command Summary

\glsxtrUpGamma \vspace{-2pt}\glsxtrUpIota \vspace{-2pt}\glsxtrUpKappa \vspace{-2pt}\glsxtrUpLambda \vspace{-2pt}\glsxtrUpMu \vspace{-2pt}\glsxtrUpNu \vspace{-2pt}\glsxtrUpOmega \vspace{-2pt}\glsxtrUpOmicron

(Sort rule) expands to the variations of math Greek upright gamma.

(Sort rule) expands to the variations of math Greek upright iota.

(Sort rule) expands to the variations of math Greek upright kappa.

(Sort rule) expands to the variations of math Greek upright lambda.

(Sort rule) expands to the variations of math Greek upright mu.

(Sort rule) expands to the variations of math Greek upright nu.

(Sort rule) expands to the variations of math Greek upright omega.

(Sort rule) expands to the variations of math Greek upright omicron.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrUpPhi</td>
<td>(Sort rule) expands to the variations of math Greek upright phi.</td>
</tr>
<tr>
<td>\glsxtrUpPi</td>
<td>(Sort rule) expands to the variations of math Greek upright pi.</td>
</tr>
<tr>
<td>\glsxtrUpPsi</td>
<td>(Sort rule) expands to the variations of math Greek upright psi.</td>
</tr>
<tr>
<td>\glsxtrUpRho</td>
<td>(Sort rule) expands to the variations of math Greek upright rho.</td>
</tr>
<tr>
<td>\glsxtrUpSigma</td>
<td>(Sort rule) expands to the variations of math Greek upright sigma.</td>
</tr>
<tr>
<td>\glsxtrUpTau</td>
<td>(Sort rule) expands to the variations of math Greek upright tau.</td>
</tr>
<tr>
<td>\glsxtrUpTheta</td>
<td>(Sort rule) expands to the variations of math Greek upright theta.</td>
</tr>
<tr>
<td>\glsxtrUpUpsilon</td>
<td>(Sort rule) expands to the variations of math Greek upright upsilon.</td>
</tr>
</tbody>
</table>
Command Summary

\texttt{\glsxtrUpXi} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright xi.

\texttt{\glsxtrUpZeta} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.27+}

(Sort rule) expands to the variations of math Greek upright zeta.

\texttt{\GlsXtrUseAbbrStyleFmts\{⟨style-name⟩\}}

\hspace{1cm} \texttt{§4.5.2; 160}

Implements the \langle\textit{display definitions}\rangle code for the given abbreviation style.

\texttt{\GlsXtrUseAbbrStyleSetup\{⟨style-name⟩\}}

\hspace{1cm} \texttt{§4.5.2; 159}

Implements the \langle\textit{setup}\rangle code for the given abbreviation style.

\texttt{\glsxtrusealias\{⟨entry-label⟩\}} \hspace{1cm} \texttt{glossaries-extra v1.42+}

\hspace{1cm} \texttt{§5.9.2; 277}

If the entry given by \langle\textit{entry-label}\rangle has the \textit{alias} field set, this will display the cross reference according to \texttt{\glsxtruseseeformat}.

\texttt{\GLSxtrusefield\{⟨entry-label⟩\}\{⟨field-label⟩\}} \hspace{1cm} \texttt{glossaries-extra v1.37+}

\hspace{1cm} \texttt{§5.11; 291}

As \texttt{\glsxtrusefield} but converts the field value to all caps.

\texttt{\Glsxtrusefield\{⟨entry-label⟩\}\{⟨field-label⟩\}} \hspace{1cm} \texttt{glossaries-extra v1.12+}

\hspace{1cm} \texttt{§5.11; 291}

As \texttt{\glsxtrusefield} but uses sentence case.
Expands to the value of the given field (identified by its internal label \langle field-label \rangle) for the entry given by \langle entry-label \rangle. Expands to \relax if the entry or field are undefined.

\glsxtruserfield

Expands to the internal label of the field used to store additional information for the “user” abbreviation styles, such as long–short–user.

\glsxtruserfieldfmt{\langle text \rangle}

Used to format the value of the field given by \glsxtruserfield within \glsxtruserparen and \GLSxtruserparen.

\GLSxtruserlongformat{\langle entry-label \rangle}{\langle fmt-cs \rangle}

As \glsxtruserlongformat but all caps.

\glsxtruserlongplformat{\langle entry-label \rangle}{\langle fmt-cs \rangle}

Formats the singular long form in parentheses (with \glsxtruserparen) in styles like short–long–user.

\GLSxtruserlongplformat{\langle entry-label \rangle}{\langle fmt-cs \rangle}

As \glsxtruserlongplformat but all caps.

\glsxtruserlongplformat{\langle entry-label \rangle}{\langle fmt-cs \rangle}

Formats the plural long form in parentheses (with \glsxtruserparen) in styles like short–long–user.
Command Summary

\GLSxtruserlongshortformat\{⟨entry-label⟩\}\{⟨insert⟩\}\{⟨long-fmt-cs⟩\}\{⟨short-fmt-cs⟩\}
glossaries-extra v1.49+

As \glsxtruserlongshortformat but all caps.

\GLSxtruserlongshortformat\{⟨entry-label⟩\}\{⟨insert⟩\}\{⟨long-fmt-cs⟩\}\{⟨short-fmt-cs⟩\}
glossaries-extra v1.49+

As \glsxtruserlongshortformat but sentence case.

\glsxtruserlongshortformat\{⟨entry-label⟩\}\{⟨insert⟩\}\{⟨long-fmt-cs⟩\}\{⟨short-fmt-cs⟩\}
glossaries-extra v1.49+

Used by styles like long–short–user to format the long and short form.

\GLSxtruserlongshortplformat\{⟨entry-label⟩\}\{⟨insert⟩\}\{⟨long-fmt-cs⟩\}\{⟨short-fmt-cs⟩\}
glossaries-extra v1.49+

As \glsxtruserlongshortplformat but all caps.

\GLSxtruserlongshortplformat\{⟨entry-label⟩\}\{⟨insert⟩\}\{⟨long-fmt-cs⟩\}\{⟨short-fmt-cs⟩\}
glossaries-extra v1.49+

As \glsxtruserlongshortplformat but sentence case.

\glsxtruserlongshortplformat\{⟨entry-label⟩\}\{⟨insert⟩\}\{⟨long-fmt-cs⟩\}\{⟨short-fmt-cs⟩\}
glossaries-extra v1.49+

Used by styles like long–short–user to format the plural long and plural short form.

\GLSxtruserparen\{⟨text⟩\}\{⟨entry-label⟩\}
glossaries-extra v1.49+
As \glsextruserparen but the value of the field given by \glsextruserfield is converted to all caps. The \textit{text} argument should already be in all caps.

\begin{verbatim}
\glsextruserparen{\langle text\rangle}{\langle entry-label\rangle}
\end{verbatim}
glossaries-extra v1.04+

Used by the “user” abbreviation styles, such as long-short-user, to insert the space separator (\glsextrfullsep) followed by the parenthetical material (\glsextrparen) consisting of \textit{text} and, if set, the value of the field given by \glsextruserfield, separated by \glsextruserparensep.

\begin{verbatim}
\glsextruserparensep
\end{verbatim}
glossaries-extra v1.49+

The separator used in the parenthetical content of \glsextruserparen and \GLSxtruserparen.

\begin{verbatim}
\GLSxtrusershortformat{\langle entry-label\rangle}{\langle fmt-cs\rangle}
\end{verbatim}
glossaries-extra v1.49+

As \glsextrusershortformat but all caps.

\begin{verbatim}
\glsextrusershortformat{\langle entry-label\rangle}{\langle fmt-cs\rangle}
\end{verbatim}
glossaries-extra v1.49+

Formats the singular short form in parentheses (with \glsextruserparen) in styles like long-short-user.

\begin{verbatim}
\GLSxtrusershortlongformat{\langle entry-label\rangle}{\langle insert\rangle}{\langle long-fmt-cs\rangle}{\langle short-fmt-cs\rangle}
\end{verbatim}
glossaries-extra v1.49+

As \glsextrusershortlongformat but all caps.

\begin{verbatim}
\GLSxtrusershortlongformat{\langle entry-label\rangle}{\langle insert\rangle}{\langle long-fmt-cs\rangle}{\langle short-fmt-cs\rangle}
\end{verbatim}
glossaries-extra v1.49+
Command Summary

As \texttt{\textbackslash glsxtrusershortlongformat} but sentence case.

\begin{verbatim}
\textbackslash glsxtrusershortlongformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle long-fmt-cs\rangle\}\{\langle short-fmt-cs\rangle\}
\end{verbatim}

Used by styles like \texttt{short-long-user} to format the short and long form.

\begin{verbatim}
\textbackslash GLSxtrusershortlongplformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle long-fmt-cs\rangle\}\{\langle short-fmt-cs\rangle\}
\end{verbatim}

As \texttt{\textbackslash glsxtrusershortlongplformat} but all caps.

\begin{verbatim}
\GLSxtrusershortlongplformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle long-fmt-cs\rangle\}\{\langle short-fmt-cs\rangle\}
\end{verbatim}

As \texttt{\textbackslash glsxtrusershortlongplformat} but sentence case.

\begin{verbatim}
\textbackslash glsxtrusershortlongplformat\{\langle entry-label\rangle\}\{\langle insert\rangle\}\{\langle long-fmt-cs\rangle\}\{\langle short-fmt-cs\rangle\}
\end{verbatim}

Used by styles like \texttt{short-long-user} to format the plural short and plural long form.

\begin{verbatim}
\textbackslash GLSxtrusershortplformat\{\langle entry-label\rangle\}\{\langle fmt-cs\rangle\}
\end{verbatim}

As \texttt{\textbackslash glsxtrusershortplformat} but all caps.

\begin{verbatim}
\textbackslash glsxtrusershortplformat\{\langle entry-label\rangle\}\{\langle fmt-cs\rangle\}
\end{verbatim}

Formats the plural short form in parentheses (with \texttt{\textbackslash glsxtruserparen}) in styles like \texttt{long-short-user}.
The plural suffix used by styles like `short-long-user`.

If the entry given by `<entry-label>` has the `see` field set, this will display the cross reference according to `\glsxtruseseeformat`.

If the entry given by `<entry-label>` has the `seealso` field set, this will display the cross reference according to `\glsxtruseseealsoformat`.

Formats the comma-separated list of entry labels as a "see also" cross-reference.

Format used by `\glsxtrusesee`. This internally uses `\glsseeformat`.

Issues a warning with `\GlossariesExtraWarning` indicating that a deprecated abbreviation style has been used.

Issues a warning with `\GlossariesExtraWarning` indicating that the given options list has been ignored by the given entry because it has already been defined.
Command Summary

\glsxtrword{⟨word⟩} \hspace{1cm} \texttt{glossaries-extra v1.17+} \hspace{1cm} §10.2.1.1; 496

Used to mark each word by the \texttt{markwords} and \texttt{markshortwords} attributes.

\glsxtrwordsep \hspace{1cm} \texttt{glossaries-extra v1.17+} \hspace{1cm} §10.2.1.1; 496

Used to mark word separator space by the \texttt{markwords} and \texttt{markshortwords} attributes.

\glsxtrwordsephyphen \hspace{1cm} \texttt{glossaries-extra v1.49+}

Used to mark compound word separator hyphen by the \texttt{markwords} and \texttt{markshortwords} attributes.

\glsxtrwrglossaryhook{⟨entry-label⟩} \hspace{1cm} \texttt{glossaries-extra v1.49+}

Hook implemented everytime an entry is indexed.

\glsxtrwrglossarylocfmt{⟨location⟩}{⟨title⟩} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.49+} \hspace{1cm} §11.5.6; 559

Used by \texttt{\glsxtrdisplaylocnameref} to format a location where the counter is wrglossary.

\glsxtrwrglosscountermark{⟨number⟩} \hspace{1cm} \texttt{glossaries-extra v1.49+} \hspace{1cm} §2.5; 29

Used to mark where the wrglossary counter is incremented with \texttt{debug=showwrgloss}.

\glsxtrwrglossmark \hspace{1cm} \texttt{glossaries-extra v1.21+} \hspace{1cm} §2.5; 29

Marker (·) used to mark write operations with \texttt{debug=showwrgloss}.
Command Summary

H

\textbf{\hyperbf{⟨location(s)⟩}}

If hyperlinks are supported this does \textbf{\hyperbf{⟨location(s)⟩}} otherwise it just does \textbf{⟨location(s)⟩}.

I

\textbf{\ifglossaryexists{⟨glossary-type⟩}{⟨true⟩}{⟨false⟩}} modifier: * glossaries

If the glossary given by ⟨glossary-type⟩ exists, this does ⟨true⟩, otherwise it does ⟨false⟩. The unstarred form treats ignored glossaries as non-existent. The starred form (v4.46+) will do ⟨true⟩ if ⟨glossary-type⟩ matches an ignored glossary.

\textbf{\ifglsentryexists{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}}

Does ⟨true⟩ if the entry given by ⟨entry-label⟩ exists, otherwise does ⟨false⟩.

\textbf{\ifglsfieldcseq{⟨entry-label⟩}{⟨field-label⟩}{⟨cs-name⟩}{⟨true⟩}{⟨false⟩}}

glossaries v4.16+

Tests if the value of the given field is equal to the replacement text of the command given by the control sequence name ⟨cs-name⟩ using etoolbox’s \ifcsstrequal. Triggers an error if the given field (identified by its internal field label) hasn’t been defined. Uses \glsdoifexists.

\textbf{\ifglsfielddefeq{⟨entry-label⟩}{⟨field-label⟩}{⟨cs⟩}{⟨true⟩}{⟨false⟩}}

glossaries v4.16+

Tests if the value of the given field is equal to the replacement text of the given command ⟨cs⟩ using etoolbox’s \ifdefstrequal. Triggers an error if the given field (identified by its internal field label) hasn’t been defined. Uses \glsdoifexists.
Command Summary

\ifglsfieldeq{⟨entry-label⟩}{⟨field-label⟩}{⟨string⟩}{⟨true⟩}{⟨false⟩}
glossaries v4.16+

Tests if the value of the given field is equal to the given string using etoolbox’s \ifcsstring. Triggers an error if the given field (identified by its internal field label) hasn’t been defined. Uses \glsdoifexists.

\ifglsfieldvoid{⟨field-label⟩}{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
glossaries v4.50+

An expandable test to determine if the entry is undefined or the field is undefined or empty. The ⟨field-label⟩ must be the field’s internal label.

\ifgls_has_children{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
glossaries v3.02+

Does ⟨true⟩ if the given entry has child entries otherwise does ⟨false⟩. Note that this has to iterate over the set of defined entries for the entry’s glossary to find one that has the entry identified in its parent field. A more efficient approach can be achieved with bib2gls and the save-child-count resource option.

\ifgls_has_desc{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
glossaries v3.08a+

Does ⟨true⟩ if the entry’s description field is set otherwise does ⟨false⟩.

\ifgls_has_desc_suppressed{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
glossaries v3.08a+

Does ⟨true⟩ if the entry’s description field is just \nopostdesc otherwise does ⟨false⟩.

\ifglsfield{⟨field⟩}{⟨entry-label⟩}{⟨true⟩}{⟨false⟩}
glossaries v4.03+

If the field identified by either its key or its internal field label ⟨field⟩ for the entry identified by ⟨entry-label⟩ is set and non-empty, this sets \glscurrentfieldvalue to the field value and does ⟨true⟩ otherwise it does ⟨false⟩.
Command Summary

\ifglshaslong\langle entry-label\rangle\{\langle true\rangle\}\{\langle false\rangle\}
glossaries v3.11a+

Does \langle true\rangle if the entry’s long field is set otherwise does \langle false\rangle.

\ifglshasparent\langle entry-label\rangle\{\langle true\rangle\}\{\langle false\rangle\}
glossaries v3.02+

Does \langle true\rangle if the entry’s parent field is set otherwise does \langle false\rangle.

\ifglshasshort\langle entry-label\rangle\{\langle true\rangle\}\{\langle false\rangle\}
glossaries v3.11a+

Does \langle true\rangle if the entry’s short field is set otherwise does \langle false\rangle.

\ifglshassymbol\langle entry-label\rangle\{\langle true\rangle\}\{\langle false\rangle\}
glossaries v3.08a+

Does \langle true\rangle if the entry’s symbol field is set otherwise does \langle false\rangle.

\ifglsindexonlyfirst \langle true\rangle\else \langle false\rangle\fi
initial: \iffalse
§5.8; 265

glossaries v3.02+

A conditional that corresponds to the indexonlyfirst option.

\ifGlsLongExtraUseTabular \langle true\rangle\else \langle false\rangle\fi
initial: \iffalse
§8.7.2; 445

glossary-longextra v1.37+

Conditional that determines whether or not to use tabular instead of longtable. If this conditional is changed, it must be changed before the style is set.

\ifglsnogroupskip \langle true\rangle\else \langle false\rangle\fi
initial: \iffalse
glossaries v3.03+

Conditional set by the nogroupskip option.
Conditional that determines whether or not to reset the entry counter to 0 when the first use flag is reset.

\ifglsresetcurrcount \langle true \rangle \else \langle false \rangle \fi

\textit{initial: \texttt{iffalse}}

\texttt{glossaries-extra v1.49+}

Does \langle true \rangle if the entry has been marked as used, does \langle false \rangle if the entry is marked as unused, and does neither if the entry hasn’t been defined (but will generate an error or warning according to \texttt{undefaction}).

\ifglsused {\langle entry-label \rangle} {\langle true \rangle} {\langle false \rangle}

\texttt{glossaries}

A conditional that indicates whether or not \texttt{wrgloss\texttt{=before}} is set.

\ifglsxtrinitwrglossbefore {\langle true \rangle} {\langle false \rangle} {\langle fi\rangle}

\texttt{glossaries-extra v1.14+}

A conditional used by the predefined abbreviation styles to determine whether the \langle insert \rangle part should go inside or outside of the style’s font formatting commands.

\ifglsxtrinsertinside {\langle true \rangle} {\langle false \rangle} {\langle fi\rangle}

\texttt{glossaries-extra v1.02}

Conditional that determines whether or not to use the last label prefix as the default.

\ifGlsXtrPrefixLabelFallbackLast {\langle true \rangle} {\langle false \rangle} {\langle fi\rangle}

\texttt{glossaries-extra-bib2gls v1.49+}

Conditional set by the \texttt{flatten} option.
Command Summary

\ifglsused\{\textit{multi-label}\}\{\texttt{true}\}\{\texttt{false}\}
\texttt{glossaries-extra v1.48+}
§7; 352

Does \texttt{true} if the given multi-entry has been marked as used, otherwise does \texttt{false}.

\ifmultiglossaryentryglobal \texttt{true}\else \texttt{false}\fi
\texttt{initial: \iffalse}
§7; 325

If true, subsequent multi-entry definitions will be global.

\IfTeXParserLib\{\texttt{\LaTeX} parser lib code\}\{\texttt{\LaTeX} code\}
\texttt{glossaries-extra-bib2gls v1.49+}
§11.5.8; 572

Defined by \texttt{glossaries-extra-bib2gls} to \texttt{\LaTeX} code but defined by \texttt{bib2gls}'s interpreter to expand to \texttt{\LaTeX} parser lib code.

\texttt{\indexname}
\texttt{initial: Index (language-sensitive)}

Expands to the index title.

\texttt{\Iota}
\texttt{glossaries-extra-bib2gls v1.27+}
§11.5.8; 573

Defined with \texttt{\providecommand}, this just does \texttt{\mathrm{I}}.

\texttt{\Kappa}
\texttt{glossaries-extra-bib2gls v1.27+}
§11.5.8; 573

Defined with \texttt{\providecommand}, this just does \texttt{\mathrm{K}}.

\texttt{\La}
\texttt{\letabbreviationstyle\{\textit{new style}\}\{\texttt{existing style}\}}
\texttt{glossaries-extra v1.04+}
§4.5.3; 161

898
Defines a synonym for an existing abbreviation style.

\loadglsentries\[\langle type\rangle]\{\langle filename\rangle\}

Locally assigns \glstextstyle to \langle type\rangle and inputs \langle filename\rangle. If the optional argument is omitted, the default glossary is assumed. Note that if any entries with \langle filename\rangle have the \texttt{type} key set (including implicitly in commands like \texttt{\newabbreviation}), then this will override the type given in the optional argument.

\longnewglossaryentry\{\langle entry-label\rangle\}\{\langle key=value list\rangle\}\{\langle description\rangle\}

Defines a new glossary entry with the given label. The second argument is a comma-separated list of glossary entry keys. The third argument is the description, which may include paragraph breaks.

\longnewglossaryentry*\{\langle entry-label\rangle\}\{\langle key=value list\rangle\}\{\langle description\rangle\}

Like the unstarrred \longnewglossaryentry but doesn’t add the \texttt{\glsxtrpostlongdescription} hook.

\MakeAcronymsAbbreviations

Counteracts \texttt{\RestoreAcronyms}. Not recommended.

\makefirstuc\{\langle text\rangle\}

Robust command that converts the first character of \langle text\rangle to uppercase unless \langle text\rangle starts with a command, in which case it will attempt to apply the case change to the first character of the first argument following the command, if the command is followed by a group. As
from mfirstuc v2.08, this command internally uses \MFUsentencecase to perform the actual case-change. See the mfirstuc documentation for further details, either:

```
texdoc mfirstuc
```
or visit ctan.org/pkg/mfirstuc.

```
\makeglossaries[⟨types⟩] glossaries
```

Opens the associated glossary files that need to be processed by makeindex or xindy. The optional argument is only available with glossaries-extra and is used for a hybrid approach. All glossaries (or each glossary identified in ⟨types⟩) should be displayed with \printglossary. If the optional argument is present, any glossaries not identified in ⟨types⟩ should be displayed with \printnoidxglossary.

```
\makenoidxglossaries glossaries v4.04+
```

Sets up all non-ignored glossaries so that they can be displayed with \printnoidxglossary.

```
\mfirstucMakeUppercase{⟨text⟩} mfirstuc
```

This command was used by \makefirstuc to convert its argument to all caps and was redefined by glossaries to use \MakeTextUppercase, but with mfirstuc v2.08+ and glossaries v4.50+ this command is instead defined to use the \LaTeX3 all caps command, which is expandable. This command is no longer used by \makefirstuc (which instead uses \MFUsentencecase) or by glossaries v4.50+ (which now uses \glsuppercase for all caps commands such as \GLS).

```
\MFUaddmap{⟨cs1⟩}{⟨cs2⟩} mfirstuc v2.08+
```

Identifies a mapping from the command ⟨cs1⟩ to command ⟨cs2⟩ for \makefirstuc and also identifies ⟨cs2⟩ as a blocker. Mappings and blockers aren’t supported by \MFUsentencecase, so both ⟨cs1⟩ and ⟨cs2⟩ are identified as exclusions for \MFUsentencecase.

```
\MFUblocker{⟨cs⟩} mfirstuc v2.08+
```

900
Command Summary

Locally identifies \langle cs \rangle as a blocker command for \makefirstuc and an exclusion for \MFUsentencecase (which doesn’t support blockers).

\MFUexcl{\langle cs \rangle}
mfirstuc v2.08+

Locally identifies \langle cs \rangle as an exclusion command, which will be recognised by both \makefirstuc and \MFUsentencecase.

\MFUsave
mfirstuc v2.08+

Saves the list of exclusions, blockers and mappings to the aux file (if required by some external tool, such as bib2gls). This command sets itself to \relax so it doesn’t repeat the action if used multiple times, and counteracts any use of \MFUsaveatend.

\MFUsaveatend
mfirstuc v2.08+

Saves the list of exclusions, blockers and mappings to the aux file (if required by some external tool, such as bib2gls) at the end of the document. This command sets itself to \relax so it doesn’t repeat the action if used multiple times, but it can be overridden by \MFUsave.

\MFUsentencecase{\langle text \rangle}
mfirstuc v2.08+

Fully expands \langle text \rangle and converts the first letter to uppercase. Unlike \makefirstuc, this command is expandable, but only recognises commands identified as exclusions. See the mfirstuc documentation for further details. This command is provided by glossaries-extra v1.49+ if an old version of mfirstuc is detected.

\MGLS[\langle options \rangle]{\langle multi-label \rangle}[\langle insert \rangle] modifiers: * + \langle alt-mod \rangle
glossaries-extra v1.48+

As \mglc but uses \GLS for each element.

\MGls[\langle options \rangle]{\langle multi-label \rangle}[\langle insert \rangle] modifiers: * + \langle alt-mod \rangle
glossaries-extra v1.48+


Command Summary

As \mgl{s} but uses \GL{s} for all elements.

\mgl{s}[⟨options⟩]{⟨multi-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩
glossaries-extra v1.48+

§7.11.1; 363

As \mgl{s} but uses \GL{s} for the first element.

\mgl{s}[⟨options⟩]{⟨multi-label⟩}[⟨insert⟩] modifiers: * + ⟨alt-mod⟩
glossaries-extra v1.48+

§7; 323

References a multi-entry identified by the given ⟨multi-label⟩.

\mgl{s}AddOptions{⟨multi-label⟩}{⟨new-options⟩}
glossaries-extra v1.48+

§7; 325

Locally appends to the options associated with the given multi-entry.

\mgl{s}Currentcategory
glossaries-extra v1.48+

§7.5; 348

Placeholder command for use in multi-entry hooks, this expands to the multi-entry category current in effect.

\mgl{s}Currentlabel
glossaries-extra v1.48+

§7.5; 348

Placeholder command for use in multi-entry hooks, this expands to the current element label.

\mgl{s}Currentlist
glossaries-extra v1.48+

§7.5; 347

Placeholder command for use in multi-entry hooks, this expands to the complete comma-separated list of elements.

\mgl{s}Currentmainlabel
glossaries-extra v1.48+

§7.5; 347
Command Summary

Placeholder command for use in multi-entry hooks, this expands to the label of the main element.

\mglscurrentmultilabel  \textit{glossaries-extra v1.48+}  \S 7.5; 347

Placeholder command for use in multi-entry hooks, this expands to the multi-entry label.

\mglscurrentoptions  \textit{glossaries-extra v1.48+}  \S 7.5; 347

Placeholder command for use in multi-entry hooks, this expands to the options used when the multi-entry was defined.

\mglscurrentprefix  \textit{glossaries-extra v1.48+}  \S 7.5; 348

Placeholder command for use in multi-entry hooks, this expands to the current prefix.

\mglscurrentsuffix  \textit{glossaries-extra v1.48+}  \S 7.5; 348

Placeholder command for use in multi-entry hooks, this expands to the current suffix.

\mglscustompostlinkhook  \textit{glossaries-extra v1.48+}  \S 7.6; 349

Hook used with the \texttt{mpostlinkelement=custom} option.

\mglscategoryprefix{\textit{category-label}}{\textit{definition}}  \textit{glossaries-extra v1.48+}  \S 7.3; 341

Defines the prefix for the given multi-entry category.

\mglscategoriesuffix{\textit{category-label}}{\textit{definition}}  \textit{glossaries-extra v1.48+}  \S 7.3; 341

Defines the suffix for the given multi-entry category.
**Command Summary**

\texttt{\textbackslash mglselementindex} \hspace{1cm} glossaries-extra v1.48+  

A count register used in multi-entry hooks, this is set to the element index.

\texttt{\textbackslash mglselementposthook} \hspace{1cm} glossaries-extra v1.48+  

Hook performed after each (non-skipped) element in a multi-entry set.

\texttt{\textbackslash mglselementprehook} \hspace{1cm} glossaries-extra v1.48+  

Hook performed before each (non-skipped) element in a multi-entry set.

\texttt{\textbackslash mglselementreset\{\langle entry-label\rangle\}} \hspace{1cm} glossaries-extra v1.48+  

Used by options such as \texttt{resetall} to reset an element’s first use flag (taking the \texttt{preset-local} option into account).

\texttt{\textbackslash mglselementunset\{\langle entry-label\rangle\}} \hspace{1cm} glossaries-extra v1.48+  

Used by options such as \texttt{unsetall} to unset an element’s first use flag (taking the \texttt{preset-local} option into account).

\texttt{\textbackslash mglsfield} \hspace{1cm} initial: useri \hspace{1cm} glossaries-extra v1.48+  

Expands to the internal field label required by \texttt{\textbackslash mglsusefield}.

\texttt{\textbackslash mglsforelements\{\langle multi-label\rangle\}\{\langle cs\rangle\}\{\langle body\rangle\}} \hspace{1cm} glossaries-extra v1.48+  

Iterates over the list of element labels for the multi-entry identified by \texttt{\langle multi-label\rangle}.

\texttt{\textbackslash mglsforotherelements\{\langle multi-label\rangle\}\{\langle cs\rangle\}\{\langle body\rangle\}} \hspace{1cm} glossaries-extra v1.48+  

Iterates over the list of element labels for the multi-entry identified by \texttt{\langle multi-label\rangle}.
Command Summary

As `\mglsforelements` but skips the main entry label.

```
\Mglsfull[⟨options⟩]⟨⟨multi-label⟩⟩[⟨insert⟩]
```

Modifiers: * + ⟨alt-mod⟩

§7.11.2; 365

glossaries-extra v1.48+

As `\mglfull` but sentence case.

```
\mglfull[⟨options⟩]⟨⟨multi-label⟩⟩[⟨insert⟩]
```

Modifiers: * + ⟨alt-mod⟩

§7.11.2; 364

glossaries-extra v1.48+

As `\mgl` but uses `\glsxtrfull` for any elements that have the `short` field set and `\glsfirst` otherwise.

```
\mglshascategoryprefix{⟨category-label⟩}⟨⟨true⟩⟩{⟨false⟩}
```

§7.3; 342

glossaries-extra v1.48+

Does ⟨true⟩ if the given multi-entry category has a prefix set otherwise does ⟨false⟩.

```
\mglshascategoriesuffix{⟨category-label⟩}⟨⟨true⟩⟩{⟨false⟩}
```

§7.3; 342

glossaries-extra v1.48+

Does ⟨true⟩ if the given multi-entry category has a suffix set otherwise does ⟨false⟩.

```
\mglsiflast⟨⟨true⟩⟩⟨⟨false⟩⟩
```

§7.5; 349

glossaries-extra v1.48+

For use in multi-entry hooks, this expands to ⟨true⟩ if this is the last iteration otherwise expands to ⟨false⟩.

```
\mglsiflastelementcapscase{⟨no-change⟩}{⟨firstuc⟩}{⟨all caps⟩}
```

§7.6.1; 351

glossaries-extra v1.48+

For use in multi-entry suffix and post-link hooks, this expands to the ⟨no-change⟩ if the last element had no case-change applied, to ⟨firstuc⟩ if the last element had sentence case applied or to ⟨all caps⟩ if the last element had all caps applied.
## Command Summary

### \texttt{\mglsiflastelements} \{\texttt{true}\}\{\texttt{false}\}

For use in multi-entry suffix and post-link hooks, this expands to the \texttt{true} if the last element was skipped, otherwise to \texttt{false}.

### \texttt{\mglsiflastelementwasfirstuse} \{\texttt{true}\}\{\texttt{false}\}

For use in multi-entry suffix and post-link hooks, this expands to the \texttt{true} if the last element was used for the first time, otherwise to \texttt{false}.

### \texttt{\mglsiflastelementwasplural} \{\texttt{true}\}\{\texttt{false}\}

For use in multi-entry suffix and post-link hooks, this expands to the \texttt{true} if the last element had the plural form displayed, otherwise to \texttt{false}.

### \texttt{\mglsiflastmaincapscase} \{\texttt{no-change}\}\{\texttt{firstuc}\}\{\texttt{all caps}\}

For use in multi-entry suffix and post-link hooks, this expands to the \texttt{no-change} if the main element from the multi-entry just referenced had no case-change applied, to \texttt{firstuc} if the last element had sentence case applied or to \texttt{all caps} if the last element had all caps applied.

### \texttt{\mglsiflastmainskipped} \{\texttt{true}\}\{\texttt{false}\}

For use in multi-entry suffix and post-link hooks, this expands to the \texttt{true} if the main element from the multi-entry just referenced was skipped, otherwise to \texttt{false}.

### \texttt{\mglsiflastmainwasfirstuse} \{\texttt{true}\}\{\texttt{false}\}

For use in multi-entry suffix and post-link hooks, this expands to the \texttt{true} if the main element from the multi-entry just referenced was used for the first time, otherwise to \texttt{false}.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\mglslastmainwasplural{true}{false}</td>
<td>For use in multi-entry suffix and post-link hooks, this expands to the (true) if the main element from the multi-entry just referenced had the plural form shown, otherwise to (false).</td>
</tr>
<tr>
<td>\mglsisfirstuse{true}{false}</td>
<td>For use in multi-entry hooks, this expands to (true) if this is the first use otherwise expands to (false).</td>
</tr>
<tr>
<td>\mglslastcategory</td>
<td>For use in multi-entry suffix and post-link hooks, this expands to the multi-entry category or nothing, if no category assigned.</td>
</tr>
<tr>
<td>\mglslastelementlabel</td>
<td>For use in multi-entry suffix and post-link hooks, this expands to the label of the last non-skipped element.</td>
</tr>
<tr>
<td>\mglslastelementpostlinkhook</td>
<td>Hook used with the mpostlinkelement=last option.</td>
</tr>
<tr>
<td>\mglslastmainlabel</td>
<td>For use in multi-entry suffix and post-link hooks, this expands to the label of the main element that was just referenced.</td>
</tr>
<tr>
<td>\mglslastmainpostlinkhook</td>
<td>Hook used with the mpostlinkelement=main option.</td>
</tr>
</tbody>
</table>

907
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\mglslastmultilabel</td>
<td>For use in multi-entry suffix and post-link hooks, this expands to the multi-entry label.</td>
</tr>
<tr>
<td>\mglsllocalreset{⟨multi-label⟩}</td>
<td>Locally resets the first use flag for the given multi-entry.</td>
</tr>
<tr>
<td>\mglsllocalunset{⟨multi-label⟩}</td>
<td>Locally unsets the first use flag for the given multi-entry.</td>
</tr>
<tr>
<td>\mglsllocalunsetothers{⟨multi-label⟩}</td>
<td>Locally unsets the first use flag for the other (not main) elements of the given multi-entry.</td>
</tr>
<tr>
<td>\Mglslong[⟨options⟩]{⟨multi-label⟩}[⟨insert⟩]</td>
<td>As \mglslong but sentence case.</td>
</tr>
<tr>
<td>\MGLSmainpl[⟨options⟩]{⟨multi-label⟩}[⟨insert⟩]</td>
<td>As \mglsl but uses \glstrlong for any elements that have the long field set and \glsttext otherwise.</td>
</tr>
</tbody>
</table>
As `\gls` but uses `\Glspl` for the main entry and `\Gls` for the others.

As `\gls` uses sentence case for the first element and the plural form for the main element.

As `\gls` but uses the plural form for the main element.

As `\gls` but uses `\Glsname`.

As `\gls` but uses `\Glsname` for the first entry and `\glsname` for the remaining entries.

As `\gls` but uses `\glsname`.
## Command Summary

As \texttt{\mgs} but uses \texttt{\Glspl} for each element.

\begin{verbatim}
\texttt{\Mglspl\langle options\rangle\{\langle multi-label\rangle\}}\[\langle insert\rangle]\text{ modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

As \texttt{\mgs} but uses \texttt{\Glspl} for each element.

\begin{verbatim}
\texttt{\Mglspl\langle options\rangle\{\langle multi-label\rangle\}}\[\langle insert\rangle]\text{ modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

As \texttt{\mgs} but uses \texttt{\Glspl} for the first element and \texttt{\glspl} for the remaining elements.

\begin{verbatim}
\texttt{\mglspl\langle options\rangle\{\langle multi-label\rangle\}}\[\langle insert\rangle]\text{ modifiers: } * + \langle alt-mod\rangle
\end{verbatim}

As \texttt{\mgs} but uses the plural form for each element.

\begin{verbatim}
\texttt{\mglsprefix}
\end{verbatim}

Code used to typeset the multi-entry prefix.

\begin{verbatim}
\texttt{\mglsreset\{\langle multi-label\rangle\}}
\end{verbatim}

Globally resets the first use flag for the given multi-entry.

\begin{verbatim}
\texttt{\mglsresetall}
\end{verbatim}

Resets the first use flag for all multi-entries.

\begin{verbatim}
\texttt{\mglsseefirstitem\{\langle multi-label\rangle\}}
\end{verbatim}
Command Summary

Formatting command used by cross-reference lists for the first item if the item is a multi-entry.

\mglsseeitem{⟨multi-label⟩} \hspace{1cm} glossaries-extra v1.48+ \hspace{1cm} §7.12; 369

Formatting command used by cross-reference lists for subsequent items if the item is a multi-entry.

\mglsSetMain{⟨multi-label⟩}⟨new-main-label⟩ \hspace{1cm} glossaries-extra v1.48+ \hspace{1cm} §7; 325

Locally changes the main element for the given multi-entry.

\mglsSetOptions{⟨multi-label⟩}⟨new-options⟩ \hspace{1cm} glossaries-extra v1.48+ \hspace{1cm} §7.11.2; 365

Locally sets the options associated with the given multi-entry.

\Mglsshort{⟨options⟩}{⟨multi-label⟩}{⟨insert⟩} \hspace{1cm} modifiers: * + {alt-mod} \hspace{1cm} glossaries-extra v1.48+ \hspace{1cm} §7.11.2; 365

As \mglsshort but sentence case.

\Mglsshort{⟨options⟩}{⟨multi-label⟩}{⟨insert⟩} \hspace{1cm} modifiers: * + {alt-mod} \hspace{1cm} glossaries-extra v1.48+ \hspace{1cm} §7.11.2; 364

As \mglss but uses \glsxtrshort for any elements that have the short field set and \gls-text otherwise.

\mglssuffix \hspace{1cm} glossaries-extra v1.48+ \hspace{1cm} §7.3; 341

Code used to typeset the multi-entry suffix.
Command Summary

\texttt{\textbackslash \textbf{MGLssymbol}}\{\langle options\rangle\}\{\langle multi-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

As \texttt{\textbackslash mgls} but uses \texttt{\textbackslash glssymbol} if the \texttt{symbol} field is set and \texttt{\textbackslash Gls} otherwise.

\texttt{\textbackslash MGLssymbol}\{\langle options\rangle\}\{\langle multi-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

As \texttt{\textbackslash mgls} but uses \texttt{\textbackslash glssymbol} if the \texttt{symbol} field is set, otherwise it uses \texttt{\textbackslash Gls} for the first element and \texttt{\textbackslash gls} for the remaining elements.

\texttt{\textbackslash MGLssymbol}\{\langle options\rangle\}\{\langle multi-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle

As \texttt{\textbackslash mgls} but uses \texttt{\textbackslash glssymbol} if the \texttt{symbol} field is set and \texttt{\textbackslash gls} otherwise.

\texttt{\textbackslash mglsunset}\{\langle multi-label\rangle\} \quad \text{glossaries-extra v1.48+}

Globally unsets the first use flag for the given multi-entry.

\texttt{\textbackslash mglsunsetall} \quad \text{glossaries-extra v1.48+}

Unsets the first use flag for all multi-entries.

\texttt{\textbackslash mglsunsetothers}\{\langle multi-label\rangle\} \quad \text{glossaries-extra v1.48+}

Globally unsets the first use flag for the other (not main) elements of the given multi-entry.

\texttt{\textbackslash mglsusecategoryprefix}\{\langle category-label\rangle\} \quad \text{glossaries-extra v1.48+}

Expands to the prefix assigned to the given multi-entry category or does nothing if no prefix assigned.
**Command Summary**

\texttt{\textbackslash mglsusecategorysuffix\{\textbackslash category-label\}\}} \quad \text{glossaries-extra v1.48+} \text{§7.3; 342}

Expands to the suffix assigned to the given multi-entry category or does nothing if no suffix assigned.

\texttt{\textbackslash MGlusefield\{\textbackslash options\}\{\textbackslash multi-label\}\{\textbackslash insert\}\}} \quad \text{modifiers: * + (alt-mod)} \text{§7.11.3; 366}

As \texttt{\textbackslash mglsusefield} but sentence case for each element.

\texttt{\textbackslash Mglusefield\{\textbackslash options\}\{\textbackslash multi-label\}\{\textbackslash insert\}\}} \quad \text{modifiers: * + (alt-mod)} \text{§7.11.3; 366}

As \texttt{\textbackslash mglsusefield} but sentence case for the first element.

\texttt{\textbackslash mglsusefield\{\textbackslash options\}\{\textbackslash multi-label\}\{\textbackslash insert\}\}} \quad \text{modifiers: * + (alt-mod)} \text{§7.11.3; 366}

As \texttt{\textbackslash mgls} but uses \texttt{\textbackslash glsdisp} if the field identified by \texttt{\textbackslash mglsfield} exists with the link text obtained from the field value.

\texttt{\textbackslash mglswasfirstuse\{\textbackslash true\}\{\textbackslash false\}\}} \quad \text{glossaries-extra v1.48+} \text{§7.6; 350}

For use in multi-entry suffix and post-link hooks, this expands to the \texttt{\{true\}} if this was the first use of the multi-entry, otherwise to \texttt{\{false\}}.

\texttt{\textbackslash MPGLS\{\textbackslash options\}\{\textbackslash multi-label\}\{\textbackslash insert\}\}} \quad \text{modifiers: * + (alt-mod)} \text{§7.11.4; 369}

As \texttt{\textbackslash mpgls} but all caps for all elements.

\texttt{\textbackslash MPGLs\{\textbackslash options\}\{\textbackslash multi-label\}\{\textbackslash insert\}\}} \quad \text{modifiers: * + (alt-mod)} \text{§7.11.4; 368}
### Command Summary

As \texttt{mpgls} but sentence case for all elements.

\[
\texttt{\textbackslash MPGLS}[\textbackslash \textit{options}][\textbackslash \textit{multi-label}][\textbackslash \textit{insert}] \quad \text{modifiers: } * + \text{ \textbackslash alt-mod}
\]

\footnotesize glossaries-extra v1.48+

§7.11.4; 368

As \texttt{mpgls} but sentence case for the first element.

\[
\texttt{\textbackslash mpGLS}[\textbackslash \textit{options}][\textbackslash \textit{multi-label}][\textbackslash \textit{insert}] \quad \text{modifiers: } * + \text{ \textbackslash alt-mod}
\]

\footnotesize glossaries-extra v1.48+

§7.11.4; 367

As \texttt{mgls} but uses \texttt{pgls} for the first element.

\[
\texttt{\textbackslash MPGLSmainpl}[\textbackslash \textit{options}][\textbackslash \textit{multi-label}][\textbackslash \textit{insert}] \quad \text{modifiers: } * + \text{ \textbackslash alt-mod}
\]

\footnotesize glossaries-extra v1.48+

§7.11.4; 369

As \texttt{mpglsmainpl} but all caps for the all elements.

\[
\texttt{\textbackslash MPGLSmainpl}[\textbackslash \textit{options}][\textbackslash \textit{multi-label}][\textbackslash \textit{insert}] \quad \text{modifiers: } * + \text{ \textbackslash alt-mod}
\]

\footnotesize glossaries-extra v1.48+

§7.11.4; 368

As \texttt{mpglsmainpl} but sentence case for all elements.

\[
\texttt{\textbackslash Mpglsmainpl}[\textbackslash \textit{options}][\textbackslash \textit{multi-label}][\textbackslash \textit{insert}] \quad \text{modifiers: } * + \text{ \textbackslash alt-mod}
\]

\footnotesize glossaries-extra v1.48+

§7.11.4; 368

As \texttt{mpglspl} but sentence case for the first element.

\[
\texttt{\textbackslash mpGLSmainpl}[\textbackslash \textit{options}][\textbackslash \textit{multi-label}][\textbackslash \textit{insert}] \quad \text{modifiers: } * + \text{ \textbackslash alt-mod}
\]

\footnotesize glossaries-extra v1.48+

§7.11.4; 368

As \texttt{mgls} but uses \texttt{pglsp1} for the first element if its the main element otherwise \texttt{pglsp} and, for the remaining elements, uses \texttt{glspl} if the element is the main entry or \texttt{glsp} otherwise.
As `\mpglspl` but all caps for the all elements.

As `\mpglspl` but sentence case for all elements.

As `\mglsp` but uses `\pglsp` for the first element and `\glspl` for the remaining elements.

Issues a warning with `\GlossariesExtraWarning` indicating that `glossaries-prefix` is required for `\mglsp` family of commands.

Defined with `\providecommand`, this just does `\mathrm{M}`.

Defines a multi-entry set with the label \texttt{(multi-label)}, consisting of the entries whose labels are listed in \texttt{(entry-label-list)}, where the main entry (which must be present in \texttt{(entry-label-list)}) is identified by \texttt{(main-label)} (or the final element in \texttt{(entry-label-list)}, if \texttt{(main-label)} is omitted).
Sets \ifmultiglossaryentryglobal to false.

\multiglossaryentryglobaltrue \texttt{glossaries-extra v1.48+} \hfill §7; 324

Sets \ifmultiglossaryentryglobal to true.

\multiglossaryentrysetup\{\texttt{⟨options⟩}\} \texttt{glossaries-extra v1.48+} \hfill §7.9; 354

Specifies a general set of options to apply to all multi-entries.

\newabbr\[\texttt{⟨options⟩}\]{\texttt{⟨entry-label⟩}}{\texttt{⟨short⟩}}{\texttt{⟨long⟩}} \hfill §4.3.2; Table 4.1

A synonym for \texttt{\newabbreviation} defined by the \texttt{shortcuts=abbreviations} or \texttt{shortcuts=ac} package option.

\newabbreviation\[\texttt{⟨options⟩}\]{\texttt{⟨entry-label⟩}}{\texttt{⟨short⟩}}{\texttt{⟨long⟩}} \hfill §4.1; 41

Defines a new entry that represents an abbreviation. This internally uses \texttt{\newglossaryentry} and any provided \texttt{⟨options⟩} (glossary entry keys) will be appended. The category is set to \texttt{abbreviation} by default, but may be overridden in \texttt{⟨options⟩}. The appropriate style should be set before the abbreviation is defined with \texttt{\setabbreviationstyle}.

\newabbreviationhook \hfill §4.1.5; 45

Hook provided within \texttt{\newabbreviation} just before the entry is defined.

\newabbreviationstyle\{\texttt{⟨style-name⟩}\}{\texttt{⟨setup⟩}}{\texttt{⟨display definitions⟩}} \hfill §4.5.3; 161

Defines an abbreviation style, which can be set with \texttt{\setabbreviationstyle}.

916
This command is provided by the base glossaries package but is redefined by glossaries-extra to use \texttt{\newabbreviation} with the \texttt{category} key set to \texttt{acronym}. The appropriate style should be set before the abbreviation is defined with \texttt{\setabbreviationstyle[acronym] {\langle style\rangle}}. You can override the \texttt{category} in \texttt{\langle options\rangle} but remember to change the optional argument of \texttt{\setabbreviationstyle} to match.

\begin{itemize}
\item \texttt{\newacronym[\langle options\rangle]{\langle entry-label\rangle}{\langle short\rangle}{\langle long\rangle}}
\end{itemize}

Defines an acronym style for use with the base glossaries package’s acronym mechanism. These styles are not compatible with glossaries-extra. Use \texttt{\newabbreviationstyle} instead.

\begin{itemize}
\item \texttt{\newacronymstyle[\langle name\rangle]{\langle format def\rangle}{\langle display def\rangle}}
\end{itemize}

Defines the command \texttt{\langle cs\rangle[\langle options\rangle]{\langle entry-label\rangle}} to behave like \texttt{\dglsfield[\langle default-options\rangle, \langle options\rangle]{\langle entry-label\rangle}{\langle field\rangle}}.

\begin{itemize}
\item \texttt{\newdglsfieldlike[\langle default-options\rangle]{\langle field\rangle}{\langle cs\rangle}{\langle Cs\rangle}{\langle CS\rangle}}
\end{itemize}

Similar to \texttt{\newdglsfield} but also defines sentence case (\texttt{\langle Cs\rangle}) and all caps (\texttt{\langle CS\rangle}) commands with mappings.

\begin{itemize}
\item \texttt{\newentry[\langle entry-label\rangle]{\langle options\rangle}}
\end{itemize}

A synonym for \texttt{\newglossaryentry} defined by the \texttt{shortcuts=other} package option.

\begin{itemize}
\item \texttt{\newglossary[\langle log-ext\rangle]{\langle glossary-label\rangle}{\langle in-ext\rangle}{\langle out-ext\rangle}{\langle title\rangle}}
\end{itemize}

Defines a glossary identified by \texttt{\langle glossary-label\rangle} (which can be referenced by the \texttt{type} key when defining an entry). The \texttt{\langle title\rangle} will be used when displaying the glossary (using commands like \texttt{\printglossary}), but this title can be overridden by the \texttt{title} option. The
optional \texttt{(counter)} indicates which counter should be used by default for the location when indexing any entries that have been assigned to this glossary. (This can be overridden by the \texttt{counter} option.) The other arguments are file extensions for use with \texttt{makeindex} or \texttt{xindy}. These arguments aren’t relevant for other indexing options (in which case, you may prefer to use \texttt{\newglossary*}).

\begin{verbatim}
\newglossary*{\langle glossary-label \rangle}{\langle title \rangle}[\langle counter \rangle]
glossaries v4.08+
\end{verbatim}

A shortcut that supplies file extensions based on the glossary label:

\begin{verbatim}
\newglossary[\langle glossary-label \rangle-glg]{\langle glossary-label \rangle}{\langle glossary-label \rangle-gls}\
{\langle glossary-label \rangle-glo}{}{\langle title \rangle}[\langle counter \rangle]
\end{verbatim}

\begin{verbatim}
\newglossaryentry{\langle entry-label \rangle}{\langle key=value list \rangle}
glossaries
\end{verbatim}

Defines a new glossary entry with the given label. The second argument is a comma-separated list of glossary entry keys.

\begin{verbatim}
\newignoredglossary{\langle glossary-label \rangle}
glossaries v4.08+
\end{verbatim}

Defines a glossary that should be ignored by iterative commands, such as \texttt{\printglossaries}. This glossary has no associated indexing files and has hyperlinks disabled. You can use an ignored glossary for common terms or abbreviations that don’t need to be included in any listing (but you may want these terms defined as entries to allow automated formatting with the \texttt{\gls}-like commands). An ignored glossary can’t be displayed with \texttt{\printglossary} but may be displayed with the “unsrt” family of commands, such as \texttt{\printunsrtglossary}.

\begin{verbatim}
\newignoredglossary*{\langle glossary-label \rangle}
glossaries-extra v1.11+
\end{verbatim}

This is like the unstarred \texttt{\newignoredglossary} but doesn’t disable hyperlinks. You will need to ensure that the hypertargets are defined. For example, with \texttt{\printunsrtglossary} or through standalone entries.

\begin{verbatim}
\newnum[\langle key=value list \rangle]{\langle entry-label \rangle}{\langle num \rangle}
\end{verbatim}

\textsection{8}; 375
\textsection{2.4}; 19
A synonym for `\glsxtrnewnumber` defined by the `shortcuts=other` package option (provided the `numbers` option is also used).

\newsym[⟨key=value list⟩]{⟨entry-label⟩}{⟨sym⟩}

A synonym for `\glsxtrnewsymbol` defined by the `shortcuts=other` package option (provided the `symbols` option is also used).

\newterm[⟨key=value list⟩]{⟨entry-label⟩}

(defines a new glossary entry with the given label, `type` set to `index`, the `name` set to `⟨entry-label⟩` and the `description` set to `\nopostdesc`. The optional argument is a comma-separated list of glossary entry keys, which can be used to override the defaults.)

\nopostdesc

When placed at the end of the `description`, this switches off the post-description hook (including the post-description punctuation). Does nothing outside of the glossary.

\Nu

`\text{N}`

Defined with `\providecommand`, this just does `\text{N}`.

\O

`\text{O}`

Defined with `\providecommand`, this just does `\text{O}`.

\omicron

`\text{O}`

Defined with `\providecommand`, this just does `\text{O}`.
Command Summary

\pagelistname \hspace{1cm} initial: Page List glossaries (language-sensitive)

Expands to the title of the location list column for headed tabular-like styles.

\PGLS\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \hspace{1cm} modifiers: * + \langle alt-mod\rangle glossaries-prefix

As \pgls but all caps.

\Pgls\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \hspace{1cm} modifiers: * + \langle alt-mod\rangle glossaries-prefix

As \pgls but sentence case.

\pgls\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \hspace{1cm} modifiers: * + \langle alt-mod\rangle glossaries-prefix

Similar to \gls but inserts the appropriate prefix, if provided.

\PGLSfmtlong\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \hspace{1cm} modifiers: * + \langle alt-mod\rangle \hspace{1cm} §5.3.2; 209

(Requires glossaries-prefix)

As \pglsfmtlong but all caps.

\Pglsfmtlong\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \hspace{1cm} modifiers: * + \langle alt-mod\rangle \hspace{1cm} §5.3.2; 209

(Requires glossaries-prefix)

As \pglsfmtlong but sentence case.

\pglsfmtlong\{\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \hspace{1cm} modifiers: * + \langle alt-mod\rangle \hspace{1cm} §5.3.2; 209

(Requires glossaries-prefix)
### Command Summary

As `\glsfmtlong` but inserts the `prefixfirst` field and separator in front if set.

\[ \text{\texttt{PGLSfmtlongpl}}\{\langle \text{options} \rangle\}\{\langle \text{entry-label} \rangle\}\{\langle \text{insert} \rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle \]

As `\glsfmtshort` but all caps.

\[ \text{\texttt{PglSfmtlongpl}}\{\langle \text{options} \rangle\}\{\langle \text{entry-label} \rangle\}\{\langle \text{insert} \rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle \]

As `\glsfmtlongpl` but sentence case.

\[ \text{\texttt{pglSfmtlongpl}}\{\langle \text{options} \rangle\}\{\langle \text{entry-label} \rangle\}\{\langle \text{insert} \rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle \]

As `\glsfmtlongpl` but inserts the `prefixfirstplural` field and separator in front if set.

\[ \text{\texttt{PGLSfmtshort}}\{\langle \text{options} \rangle\}\{\langle \text{entry-label} \rangle\}\{\langle \text{insert} \rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle \]

As `\glsfmtshort` but all caps.

\[ \text{\texttt{PglSfmtshort}}\{\langle \text{options} \rangle\}\{\langle \text{entry-label} \rangle\}\{\langle \text{insert} \rangle\} \quad \text{modifiers: } * + \langle \text{alt-mod} \rangle \]

As `\glsfmtshort` but sentence case.

\[ \]
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Example</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\pglsfmtshort</code></td>
<td>As <code>\glsfmtshort</code> but inserts the <code>prefix</code> field and separator in front if set.</td>
<td>§5.3.2; 207</td>
<td></td>
</tr>
<tr>
<td><code>\PGLSfmtshortpl</code></td>
<td>As <code>\pglsfmtshortpl</code> but all caps.</td>
<td>§5.3.2; 208</td>
<td></td>
</tr>
<tr>
<td><code>\Pglsfmtshortpl</code></td>
<td>As <code>\pglsfmtshortpl</code> but sentence case.</td>
<td>§5.3.2; 208</td>
<td></td>
</tr>
<tr>
<td><code>\pglsfmtshortpl</code></td>
<td>As <code>\glsfmtshortpl</code> but inserts the <code>prefixplural</code> field and separator in front if set.</td>
<td>§5.3.2; 208</td>
<td></td>
</tr>
<tr>
<td><code>\PGLSpl</code></td>
<td>As <code>\pglspl</code> but all caps.</td>
<td>§5.3.2; 208</td>
<td></td>
</tr>
<tr>
<td><code>\Pglspl</code></td>
<td>As <code>\pgls</code> but sentence case.</td>
<td>§5.3.2; 208</td>
<td></td>
</tr>
</tbody>
</table>
Command Summary

\texttt{\textbackslash pglspl}[\langle \text{options} \rangle \{\langle \text{entry-label} \rangle \} \{\langle \text{insert} \rangle \}]
\text{ modifiers: } * + \langle \text{alt-mod} \rangle
\texttt{glossaries\text{-}prefix}

Similar to \texttt{\textbackslash glspl} but inserts the appropriate prefix, if provided.

\texttt{\textbackslash PGLSxtrlong}[\langle \text{options} \rangle \{\langle \text{entry-label} \rangle \} \{\langle \text{insert} \rangle \}]
\text{ modifiers: } * + \langle \text{alt-mod} \rangle
\texttt{glossaries\text{-}extra v1.49+}
(requires glossaries\text{-}prefix)

As \texttt{\textbackslash pglsxtrlong} but all caps.

\texttt{\textbackslash Pglsxtrlong}[\langle \text{options} \rangle \{\langle \text{entry-label} \rangle \} \{\langle \text{insert} \rangle \}]
\text{ modifiers: } * + \langle \text{alt-mod} \rangle
\texttt{glossaries\text{-}extra v1.49+}
(requires glossaries\text{-}prefix)

As \texttt{\textbackslash pglsxtrlong} but sentence case.

\texttt{\textbackslash pglsxtrlong}[\langle \text{options} \rangle \{\langle \text{entry-label} \rangle \} \{\langle \text{insert} \rangle \}]
\text{ modifiers: } * + \langle \text{alt-mod} \rangle
\texttt{glossaries\text{-}extra v1.49+}
(requires glossaries\text{-}prefix)

As \texttt{\textbackslash glsxlsxtrlong} but inserts the \texttt{prefixfirst} field and separator in front if set.

\texttt{\textbackslash PGLSxtrlongpl}[\langle \text{options} \rangle \{\langle \text{entry-label} \rangle \} \{\langle \text{insert} \rangle \}]
\text{ modifiers: } * + \langle \text{alt-mod} \rangle
\texttt{glossaries\text{-}extra v1.49+}
(requires glossaries\text{-}prefix)

As \texttt{\textbackslash pglsxtrlongpl} but all caps.

\texttt{\textbackslash Pglsxtrlongpl}[\langle \text{options} \rangle \{\langle \text{entry-label} \rangle \} \{\langle \text{insert} \rangle \}]
\text{ modifiers: } * + \langle \text{alt-mod} \rangle
\texttt{glossaries\text{-}extra v1.49+}
(requires glossaries\text{-}prefix)

As \texttt{\textbackslash pglsxtrlongpl} but sentence case.
Command Summary

\texttt{\textbackslash pglsxtrlongpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} modifiers: * + ⟨alt-mod⟩

As \texttt{\textbackslash glsxtrlongpl} but inserts the \texttt{prefixfirstplural} field and separator in front if set.

\texttt{\textbackslash PGLSxtrshort[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} modifiers: * + ⟨alt-mod⟩

As \texttt{\textbackslash pglsxtrshort} but all caps.

\texttt{\textbackslash Pglsxtrshort[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} modifiers: * + ⟨alt-mod⟩

As \texttt{\textbackslash pglsxtrshort} but sentence case.

\texttt{\textbackslash pglsxtrshort[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} modifiers: * + ⟨alt-mod⟩

As \texttt{\textbackslash glsxtrshort} but inserts the \texttt{prefix} field and separator in front if set.

\texttt{\textbackslash PGLSxtrshortpl[⟨options⟩]{⟨entry-label⟩}[⟨insert⟩]} modifiers: * + ⟨alt-mod⟩

As \texttt{\textbackslash pglsxtrshortpl} but all caps.
### Command Summary

As `\glsxtrshortpl` but sentence case.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Requires</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glsxtrshortpl</code> [⟨options⟩] {⟨entry-label⟩} [⟨insert⟩] modifiers: * + ⟨alt-mod⟩</td>
<td>As <code>\glsxtrshortpl</code> but inserts the <code>prefixplural</code> field and separator in front if set.</td>
<td>glossaries-extra v1.49+</td>
<td>§4.3.1; 55</td>
</tr>
</tbody>
</table>

As `\glsxtrtitlelong`.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Requires</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glsxtrtitlelong</code> {⟨entry-label⟩}</td>
<td>The normal behaviour of <code>\glsxfmtlong</code>.</td>
<td>glossaries-extra v1.49+</td>
<td>§5.3.3; 219</td>
</tr>
</tbody>
</table>

The normal behaviour of `\glsxtrtitlelongpl`.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Requires</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\glsxtrtitleshort</code> {⟨entry-label⟩}</td>
<td>The normal behaviour of <code>\glsxfmtshort</code>.</td>
<td>glossaries-extra v1.49+</td>
<td>§5.3.3; 219</td>
</tr>
</tbody>
</table>

The normal behaviour of `\glsxtrtitleshortpl`.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Requires</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\pretoglossarypreamble</code> [⟨type⟩] {⟨text⟩}</td>
<td>Prepends (locally) ⟨text⟩ to the preamble for the glossary identified by ⟨type⟩. If ⟨type⟩ is omitted, <code>\glsdefaulttype</code> is assumed.</td>
<td>glossaries-extra v1.12+</td>
<td>§8.2; 377</td>
</tr>
</tbody>
</table>
\texttt{\textbackslash printabbreviations[(options)]]} (requires \texttt{\usepackage[abbreviations]{glossaries-extra}})

Shortcut for \texttt{\textbackslash printglossary[type=glsxtrabbrvtype]}.

\texttt{\textbackslash printacronyms[(options)]} (requires the \texttt{acronyms} package option)

Shortcut for \texttt{\textbackslash printglossary[type=acronymtype]}.

\texttt{\textbackslash printglossaries} (requires \texttt{glossaries})

Iterates over all non-ignored glossaries and does \texttt{\textbackslash printglossary[type=(type)]} for each glossary.

\texttt{\textbackslash printglossary[(options)]} (requires \texttt{glossaries})

Displays the glossary by inputting a file created by \texttt{makeindex} or \texttt{xindy}. Must be used with \texttt{\makeglossaries} and either \texttt{makeindex} or \texttt{xindy}.

\texttt{\textbackslash printindex[(options)] v4.02+} (requires the \texttt{index} package option)

Shortcut provided by the \texttt{index} package option that simply does \texttt{\textbackslash printglossary[type=index]}.

\texttt{\textbackslash printnoidxglossaries} (requires \texttt{glossaries v4.04+})

Iterates over all non-ignored glossaries and does \texttt{\textbackslash printnoidxglossary[type=(type)]} for each glossary.

\texttt{\textbackslash printnoidxglossary[(options)]} (requires \texttt{glossaries v4.04+})
Displays the glossary by obtaining the indexing information from the aux file and using \TeX to sort and collate. Must be used with \texttt{\makenoidxglossaries} or with the glossaries not identified in the optional argument of \texttt{\makeglossaries} when using the hybrid method. This method can be very slow and has limitations.

\texttt{\textbackslash printnumbers[⟨options⟩]} \hspace{1cm} \texttt{\textbackslash printglossary[type=numbers]} \hspace{1cm} \texttt{glossaries v4.02+}

Shortcut for \texttt{\textbackslash printglossary[type=numbers]}.

\texttt{\textbackslash printsymbols[⟨options⟩]} \hspace{1cm} \texttt{\textbackslash printglossary[type=symbols]} \hspace{1cm} \texttt{glossaries v4.02+}

Shortcut for \texttt{\textbackslash printglossary[type=symbols]}.

\texttt{\textbackslash printunsrtabbreviations[⟨options⟩]} \hspace{1cm} \texttt{\textbackslash printunsrtglossary[type=\textbackslash gls\textbackslash x\textbackslash abbrev\textbackslash r\textbackslash type]} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.40+}

Shortcut for \texttt{\textbackslash printunsrtglossary[type=\textbackslash gls\textbackslash x\textbackslash abbrev\textbackslash r\textbackslash type]}.

\texttt{\textbackslash printunsrtacronyms[⟨options⟩]} \hspace{1cm} \texttt{\textbackslash printunsrtglossary[type=\textbackslash acronym\textbackslash type]} \hspace{1cm} \texttt{glossaries-extra-bib2gls v1.40+}

Shortcut for \texttt{\textbackslash printunsrtglossary[type=\textbackslash acronym\textbackslash type]}.

\texttt{\textbackslash printunsrtglossaries} \hspace{1cm} \texttt{\textbackslash printunsrtglossary[type=(\textbackslash acronym\textbackslash type)]} \hspace{1cm} \texttt{glossaries-extra v1.08+}

Iterates over all non-ignored glossaries and does \texttt{\textbackslash printunsrtglossary[type=(\textbackslash acronym\textbackslash type)]} for each glossary.
Command Summary

Displays the glossary by iterating over all entries associated with the given glossary (in the order in which they were added to the glossary). Group headers will only be inserted if the `group` key has been defined and has been set (typically with the `record` option and `bib2gls`). Location lists will only be shown if the `location` or `loclist` fields have been set (typically by `bib2gls`).

\texttt{\textbackslash \texttt{printunsrtglossary*}[\langle options\rangle]\{\langle init-code\rangle\}} \quad \text{glossaries-extra v1.12+} \quad \S 8.4; 383

Does \{\langle init-code\rangle\texttt{\textbackslash \texttt{printunsrtglossary}[\langle options\rangle]\}} which localises \langle init-code\rangle.

\texttt{\textbackslash \texttt{printunsrtglossaryentryprocesshook}\{\langle entry-label\rangle\}} \quad \text{glossaries-extra v1.21+} \quad \S 8.4.3; 401

Hook used within the “unsrt” family of commands while the glossary is being constructed.

\texttt{\textbackslash \texttt{printunsrtglossaryhandler}\{\langle entry-label\rangle\}} \quad \text{glossaries-extra v1.16+} \quad \S 8.4.3; 402

Used within the “unsrt” family of commands to process the current entry.

\texttt{\textbackslash \texttt{printunsrtglossarypredoglossary}} \quad \text{glossaries-extra v1.21+} \quad \S 8.4.3; 401

Hook performed by the “unsrt” family of commands just before the glossary body is displayed.

\texttt{\textbackslash \texttt{printunsrtglossaryskipentry}} \quad \text{glossaries-extra v1.21+} \quad \S 8.4.3; 401

May be used within \texttt{\textbackslash \texttt{printunsrtglossaryentryprocesshook} to skip the current entry.}

\texttt{\textbackslash \texttt{printunsrtglossaryunit\[\langle options\rangle\\\{\langle counter-name\rangle\}}} \quad \text{glossaries-extra v1.12+} \quad \S 8.4.3.2; 408

Provided for use with \texttt{\textbackslash GlsXtrRecordCounter} to display a glossary with \texttt{\textbackslash \texttt{printunsrtglossary*} that filters entries that don’t have a match for the current \langle counter-name\rangle value.
**Command Summary**

\texttt{\textbackslash printunsrtglossaryunitpostskip} \texttt{glossaries-extra v1.49+} \texttt{§8.4.3.2; 409}

The vertical space at the end of the glossary appended by \texttt{\textbackslash printunsrtglossaryunit}.

\texttt{\textbackslash printunsrtglossaryunitsetup\{\langle counter-name\rangle\}} \texttt{glossaries-extra v1.12+} \texttt{§8.4.3.2; 409}

Sets up the filtering used by \texttt{\textbackslash printunsrtglossaryunit}.

\texttt{\textbackslash printunsrtindex\{\langle options\rangle\}} \texttt{glossaries-extra-bib2gls v1.40+} \texttt{(requires \textbackslash usepackage\{index,record\}\{glossaries-extra\})} \texttt{§11.5.1; 535}

Shortcut provided by the index package option combined with glossaries-extra-bib2gls that simply does \texttt{\textbackslash printunsrtglossary[type=index]}.

\texttt{\textbackslash printunsrtinnerglossary\{\langle options\rangle\}\{\langle pre-code\rangle\}\{\langle post-code\rangle\}} \texttt{glossaries-extra v1.44+} \texttt{§8.4.3.1; 403}

Similar to \texttt{\textbackslash printunsrtglossary} but doesn’t contain the code that starts and ends the glossary (such as beginning and ending the theglossary environment), so this command needs to be either placed inside printunsrtglossarywrap or in the \texttt{\textbackslash printunsrtglossary} entry handler \texttt{\textbackslash printunsrtglossaryhandler}.

\texttt{\textbackslash printunsrtnumbers\{\langle options\rangle\}} \texttt{glossaries-extra-bib2gls v1.40+} \texttt{(requires \textbackslash usepackage\{numbers,record\}\{glossaries-extra\})} \texttt{§11.5.1; 534}

Shortcut provided by the numbers package option combined with glossaries-extra-bib2gls that simply does \texttt{\textbackslash printunsrtglossary[type=numbers]}.

\texttt{\textbackslash printunsrtsymbols\{\langle options\rangle\}} \texttt{glossaries-extra-bib2gls v1.40+} \texttt{(requires \textbackslash usepackage\{symbols,record\}\{glossaries-extra\})} \texttt{§11.5.1; 534}

Shortcut for \texttt{\textbackslash printunsrtglossary[type=symbols]}.
Command Summary

\printunsrttable\[\langle options\rangle]\quad \text{glossary-table v1.49+}

§8.7.4; 468

Internally uses \printunsrtglossary with the table style.

\provideignoredglossary\{\langle glossary-label\rangle\} \quad \text{modifier: * glossaries-extra v1.12+}

§8; 376

As \newignoredglossary but does nothing if the glossary has already been defined.

\providemultiglossaryentry\[\langle options\rangle\]\{\langle multi-label\rangle\}\{\langle main-label\rangle\}\{\langle entry-label-list\rangle\}

\text{glossaries-extra v1.48+}

§7; 324

As \multiglossaryentry but does nothing if a multi-entry set has already been defined with the given label.

\ProvidesGlossariesExtraLang\{\langle tag\rangle\}

§15; 593

Should be placed at the start of a glossaries-extra ldf file.

R

\renewabbreviationstyle\{\langle style-name\rangle\}\{\langle setup\rangle\}\{\langle display definitions\rangle\}

\text{glossaries-extra v1.04+}

§4.5.3; 161

Redefines an abbreviation style.

\RequireGlossariesExtraLang\{\langle tag\rangle\}

Indicates that a glossaries-extra ldf file should be input, if it hasn’t already been input.

\RestoreAcronyms

§4.6; 181

Restores \newacronym to the base glossaries mechanism. Not recommended.
Command Summary

\rGLS\[\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle

Like \textbackslash GLS but hooks into the entry’s record count.

\rGls\[\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle

Like \textbackslash Gls but hooks into the entry’s record count.

\rgls\[\langle options\rangle\}\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle

Like \textbackslash gls but hooks into the entry’s record count.

\rGLSformat\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{glossaries-extra v1.21+}

Format used by \textbackslash rGLS if the entry’s record count is more than the given trigger value.

\rGlsformat\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{glossaries-extra v1.21+}

Format used by \textbackslash rGls if the entry’s record count is more than the given trigger value.

\rglsformat\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{glossaries-extra v1.21+}

Format used by \textbackslash rgls if the entry’s record count is more than the given trigger value.

\rGLSp1\[\langle options\rangle\]\{\langle entry-label\rangle\}\{\langle insert\rangle\} \quad \text{modifiers: } * + \langle alt-mod\rangle

Like \textbackslash GLSp1 but hooks into the entry’s record count.
### Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glspl{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Like \Glspl{} but hooks into the entry’s record count.</td>
<td>glossaries-extra v1.21+</td>
</tr>
<tr>
<td>\rglspl{⟨options⟩}{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Like \glspl{} but hooks into the entry’s record count.</td>
<td>glossaries-extra v1.21+</td>
</tr>
<tr>
<td>\GLSplformat{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Format used by \GLSpl{} if the entry’s record count is more than the given trigger value.</td>
<td>glossaries-extra v1.21+</td>
</tr>
<tr>
<td>\Glsplformat{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Format used by \Glspl{} if the entry’s record count is more than the given trigger value.</td>
<td>glossaries-extra v1.21+</td>
</tr>
<tr>
<td>\rglsplformat{⟨entry-label⟩}{⟨insert⟩}</td>
<td>Format used by \rglspl{} if the entry’s record count is more than the given trigger value.</td>
<td>glossaries-extra v1.21+</td>
</tr>
<tr>
<td>\Rho</td>
<td>Defined with \providecommand, this just does \mathrm{P}.</td>
<td>glossaries-extra-bib2gls v1.27+</td>
</tr>
</tbody>
</table>

**S**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>\seealso{⟨insert⟩}</td>
<td>(language-sensitive)</td>
<td>glossaries-extra v1.16+</td>
</tr>
</tbody>
</table>

932
## Command Summary

Used as a cross-reference tag. The default value is `\alsoname`, if that command has been defined, or “see also”.

\begin{center}  \texttt{\seename} \hspace{1cm} \textit{initial: see} \hspace{1cm} \text{(language-sensitive)} \end{center}

Used as a cross-reference tag (provided by glossaries if not already defined).

\begin{center}  \texttt{\setabbreviationstyle[\langle category \rangle]{\langle style-name \rangle}} \hspace{1cm} \text{\S 4.5; 59} \end{center}

Sets the current abbreviation style to `\langle style-name \rangle` for the category identified by `\langle category \rangle`. If the optional argument is omitted, `abbreviation` is assumed.

\begin{center}  \texttt{\setacronymstyle{\langle acronym-style-name \rangle}} \hspace{1cm} \text{glossaries v4.02+} \end{center}

Sets the style for the base glossaries package’s acronym mechanism. These styles are not compatible with glossaries-extra, which redefines `\newacronym` to use `\newabbreviation`.

Use:

\begin{center}  \texttt{\setabbreviationstyle[acronym]{\langle abbreviation-style-name \rangle}} \end{center}

with the closest matching abbreviation style instead.

\begin{center}  \texttt{\setentrycounter[\langle prefix \rangle]{\langle counter \rangle}} \hspace{1cm} \text{glossaries} \end{center}

Used to set the location counter and prefix required for `\glshypernumber`.

\begin{center}  \texttt{\setglossarypreamble[\langle type \rangle]{\langle text \rangle}} \hspace{1cm} \text{glossaries v3.07+} \end{center}

Globally sets the preamble for the glossary identified by `\langle type \rangle` to `\langle text \rangle`. If `\langle type \rangle` is omitted, `\glsdefaulttype` is assumed.

\begin{center}  \texttt{\setglossarystyle{\langle style-name \rangle}} \hspace{1cm} \text{glossaries v3.08a+} \end{center}
Set the current glossary style to \langle style-name \rangle. Redefined by glossaries-extra to include \glsxtrpreglossarystyle.

\setupglossaries\langle options \rangle \hspace{1.5cm} \text{glossaries v3.11a+} \hspace{1cm} \S2; 9

Change allowed options that are defined by the base glossaries package. Note that some options can only be passed as package options. To change options defined or modified by the glossaries-extra package, use \glossariesextrasetup.

\setupglsadd\langle options \rangle \hspace{1.5cm} \text{glossaries-extra v1.49+} \hspace{1cm} \S5.1.1; 190

Set the default options for \glsadd.

\setupglslink\langle options \rangle \hspace{1.5cm} \text{glossaries-extra v1.49+} \hspace{1cm} \S5.1.1; 190

Set the default \glslink options.

\subglossentry\langle level \rangle\langle entry label \rangle\langle location list \rangle \hspace{1.5cm} \text{glossaries v3.08+}

Used to format a child entry. This command should be redefined by the glossary style.

\symbolname \hspace{1.5cm} \text{initial: Symbol} \hspace{1.5cm} \text{glossaries}

Expands to the title of the symbol column for headed tabular-like styles.

\T

\Tau \hspace{1.5cm} \text{glossaries-extra-bib2gls v1.27+} \hspace{1cm} \S11.5.8; 573

Defined with \providecommand, this just does \text{\texttt{\textbackslash{}mathrm{T}}}.
Command Summary

\texttt{\textbackslash theglossaryentry} \glossaries v3.0+

The value of the glossaryentry counter.

\texttt{U}

\texttt{\textbackslash u\langle hex\rangle}

Recognised by \texttt{bib2gls} within some resource options as identifying the Unicode character given by \langle hex\rangle. Since \texttt{\textbackslash u} is defined by the \TeX kernel, you need to protect it from expansion while the options are written to the aux file (\texttt{\string\textbackslash u\langle hex\rangle}).

\texttt{\textbackslash Upalpha} \glossaries-extra-bib2gls v1.27+

Defined with \texttt{\providecommand} and only if \texttt{upgreek} has been loaded, this just does \texttt{\mathsf{A}}.

\texttt{\textbackslash Upbeta} \glossaries-extra-bib2gls v1.27+

Defined with \texttt{\providecommand} and only if \texttt{upgreek} has been loaded, this just does \texttt{\mathsf{B}}.

\texttt{\textbackslash Upchi} \glossaries-extra-bib2gls v1.27+

Defined with \texttt{\providecommand} and only if \texttt{upgreek} has been loaded, this just does \texttt{\mathsf{X}}.

\texttt{\textbackslash Upepsilon} \glossaries-extra-bib2gls v1.27+

Defined with \texttt{\providecommand} and only if \texttt{upgreek} has been loaded, this just does \texttt{\mathsf{E}}.

\texttt{\textbackslash Upeta} \glossaries-extra-bib2gls v1.27+

935
Defined with `\providecommand` and only if `upgreek` has been loaded, this just does `\mathrm{H}`.

\[ \Upiota \]

Defined with `\providecommand` and only if `upgreek` has been loaded, this just does `\mathrm{I}`.

\[ \Upkappa \]

Defined with `\providecommand` and only if `upgreek` has been loaded, this just does `\mathrm{K}`.

\[ \Upmu \]

Defined with `\providecommand` and only if `upgreek` has been loaded, this just does `\mathrm{M}`.

\[ \Upnu \]

Defined with `\providecommand` and only if `upgreek` has been loaded, this just does `\mathrm{N}`.

\[ \Upomicron \]

Defined with `\providecommand` and only if `upgreek` has been loaded, this just does `\mathrm{o}`.

\[ \upomicron \]
Command Summary

\Uprho  
Defined with \providecommand and only if upgreek has been loaded, this just does \mathrm \{P\}.

\Uptau  
Defined with \providecommand and only if upgreek has been loaded, this just does \mathrm \{T\}.

\Upzeta  
Defined with \providecommand and only if upgreek has been loaded, this just does \mathrm \{Z\}.

\maketitle

W
\texttt{\writemultiglossentry\{\textbackslash options\}\{\textbackslash multi-label\}\{\textbackslash main-label\}\{\textbackslash list\}}
glossaries-extra v1.48+

Writes multi-entry information to the aux file.

X
\texttt{\xcapitalisefmtwords\{\textbackslash text\}}
mfirstuc v2.03+

Passes the argument to \capitalisefmtwords but with the first token in \textbackslash text expanded. The starred version uses the starred version of \capitalisefmtwords.

\texttt{\xglssetwidest\{\textbackslash level\}\{\textbackslash name\}}
glossaries-extra-stylemods v1.05+

As \xglssetwidest but global.
**Command Summary**

\texttt{\textbackslash xglsupdatewidest[\langle level\rangle][\langle name\rangle]}  
\textit{glossaries-extra-stylemods v1.23+}  
§8.6.5.4; 434

As \texttt{\textbackslash eglsupdatewidest} but global.

\texttt{\textbackslash xGlsXtrIfValueInFieldCsvList\{\langle entry-label\rangle\}{\langle field-label\rangle}\{\langle value\rangle\}{\langle true\rangle}\{\langle false\rangle\}}  
\textit{modifier: *}  
\textit{glossaries-extra v1.47+}  
§5.13; 304

As \texttt{\textbackslash GlsXtrIfValueInFieldCsvList} but fully expands \langle value\rangle.

\texttt{\textbackslash xGlsXtrSetField\{\langle entry-label\rangle\}{\langle field-label\rangle}\{\langle value\rangle\}}  
\textit{glossaries-extra v1.12+}  
§3.5; 40

As \texttt{\textbackslash GlsXtrSetField} but expands the value and uses a global assignment.

\texttt{\textbackslash xpglsxtrpostabbrvfootnote}  
\textit{glossaries-extra v1.49+}  
§4.5.2; 160

Expands to the footnote code required for styles like \texttt{short-postfootnote}.

\texttt{\textbackslash xpglsxtrposthyphenlong}  
\textit{glossaries-extra v1.49+}  
152

Used within the post-link hook for the \texttt{short-hyphen-postlong-hyphen} style on first use. This expands the placeholder commands and uses either \texttt{\textbackslash glsxtrposthyphenlong} or \texttt{\textbackslash GLSxtr-posthyphenlong}.

\texttt{\textbackslash xpglsxtrposthyphenshort}  
\textit{glossaries-extra v1.49+}  
150

Used within the post-link hook for the \texttt{long-hyphen-postshort-hyphen} style on first use. This expands the placeholder commands and uses either \texttt{\textbackslash glsxtrposthyphenshort} or \texttt{\textbackslash GLSxtr-posthyphenshort}.

\texttt{\textbackslash xpglsxtrposthyphensubsequent}  
\textit{glossaries-extra v1.49+}  
151

938
Command Summary

Used within the post-link hook for the `long-hyphen-postshort-hyphen` style on subsequent use. This expands the placeholder commands and uses either \glsxtrposthyphensubsequent or \GLSxtrposthyphensubsequent.

\[ \Zeta \]

Defined with `\providecommand`, this just does `\mathrm{\Zeta}`.

\section{Glossaries-Extra-Bib2Gls v1.27+}

\chapter{939}
Environment Summary

\begin{printunsrtglossarywrap}[⟨options⟩] glossaries-extra v1.44+

Sets up the start and end of the glossary (including beginning and ending the \texttt{theglossary} environment). Use \texttt{printunsrtinnerglossary} within the body for each block of entries.

\begin{theglossary}
\end{theglossary}

Redefined by glossary styles to set up the way the glossary is displayed. For example, to start and end the description environment for the list styles.
Package Option Summary

\usepackage[⟨options⟩]{glossaries-extra-stylemods}  
   (or \usepackage[stylemods=⟨options⟩]{glossaries-extra})

Modifies the glossary styles supplied with the base glossaries package to make them more flexible and to integrate support for features provided by glossaries-extra or bib2gls.

all
Load all predefined styles.

⟨name⟩
Load package glossary-⟨name⟩.

\usepackage[⟨options⟩]{glossaries-extra}

Extension package that loads glossaries, provides additional commands, and modifies some of the base glossaries commands to integrate them with the new commands or to make them more flexible.

abbreviations
Provides a new glossary with the label abbreviations and title given by \abbreviations-name, redefines \glsxtrabbrvtype to abbreviations, redefines \acronymtype to \glsxtrabbrvtype (unless the acronym or acronyms option has been used), and provides \printabbreviations.

accsupp
 Loads glossaries-accsupp.

autoseeindex=⟨boolean⟩  \hspace{1em} default: true; initial: true
Indicates whether or not to enable automatic indexing of see andseealso fields.

autoseeindex=false
Disables automatic indexing of see andseealso fields.

autoseeindex=true
Enables automatic indexing of see andseealso fields.
Package Option Summary

\texttt{bibglsaux=\langle basename\rangle}

\hspace{1cm} \textit{initial: empty} \hspace{1cm} glossaries-extra v1.49+

(\textit{requires bib2gls v3.0+})

If set, an additional \texttt{aux} file called \texttt{\langle basename\rangle.aux} will be created in which to store the \texttt{bib2gls} records. This file will be skipped by \LaTeX{} when the main \texttt{aux} file is input but will be read by \texttt{bib2gls}.

\texttt{debug=(value)}

\hspace{1cm} \textit{default: true; initial: false} \hspace{1cm} §2.5; 28

Provides debugging information. Some options are also available with just the base \texttt{glossaries} package.

\begin{itemize}
\item \texttt{debug=all} glossaries-extra v1.21+
\hspace{1cm} Implements all debugging options.
\item \texttt{debug=false} glossaries v4.24+
\hspace{1cm} Disable debugging actions.
\item \texttt{debug=showaccsupp} glossaries v4.45+
\hspace{1cm} Implements \texttt{debug=true} and also shows accessibility information in the document.
\item \texttt{debug=showtargets} glossaries v4.24+
\hspace{1cm} Implements \texttt{debug=true} and also shows target markers in the document.
\item \texttt{debug=showwrgloss} glossaries-extra v1.21+
\hspace{1cm} Implements \texttt{debug=true} and also shows a marker in the document whenever the entry is indexed and (if used with \texttt{indexcounter}) will mark where the \texttt{wrglossary} counter is incremented.
\item \texttt{debug=true} glossaries v4.24+
\hspace{1cm} Writes \texttt{wrglossary(\langle type\rangle)(\langle indexing info\rangle)} to the log file if there is an attempt to index an entry before the associated indexing file has been opened (\texttt{makeindex} and \texttt{xindy} only). With \texttt{glossaries-extra}, this setting will also display the label of any undefined entries that are referenced in the document.
\end{itemize}

\texttt{docdef=(value)}

\hspace{1cm} \textit{default: true; initial: false} \hspace{1cm} §2.4; 16

Determines whether or not \texttt{\newglossaryentry} is permitted in the document environment.

\begin{itemize}
\item \texttt{docdef=atom}
\hspace{1cm} As restricted but creates the \texttt{glsdefs} file for atom’s autocomplete support.
\item \texttt{docdef=false}
\hspace{1cm} Don’t allow \texttt{\newglossaryentry} in the document environment.
\item \texttt{docdef=restricted}
\hspace{1cm} Allow \texttt{\newglossaryentry} in the document environment, but only before any glossaries.
\item \texttt{docdef=true}
\end{itemize}
Package Option Summary

Allow \texttt{\textbackslash newglossaryentry} in the document environment if the base glossaries package would allow it.

\begin{itemize}
  \item \texttt{equations=} \texttt{(boolean)} \hspace{1cm} \textit{default: true; initial: false} \hspace{1cm} \textsection{2.4; 26}
    Automatically switch the location counter to equation when inside a numbered equation environment.

  \item \texttt{floats=} \texttt{(boolean)} \hspace{1cm} \textit{default: true; initial: false} \hspace{1cm} \textsection{2.4; 27}
    Automatically switch the location counter to the corresponding counter when inside a floating environment.

  \item \texttt{indexcounter} \hspace{1cm} \textsection{2.4; 27}
    Defines the index counter \texttt{wrglossary} and implements \texttt{counter=wrglossary}.

  \item \texttt{indexcrossrefs=} \texttt{(boolean)} \hspace{1cm} \textit{default: true; initial: varies} \hspace{1cm} \textsection{2.4; 20}
    Indicates whether or not to enable automatic indexing of cross-referenced entries.

    \begin{itemize}
      \item \texttt{indexcrossrefs=false} \hspace{1cm} \textsection{21}
        Disables automatic indexing of cross-referenced entries.

      \item \texttt{indexcrossrefs=true} \hspace{1cm} \textsection{21}
        Enables automatic indexing of cross-referenced entries.
    \end{itemize}

  \item \texttt{nomissingglistext=} \texttt{(boolean)} \hspace{1cm} \textit{default: true; initial: false} \hspace{1cm} \textsection{2.1; 9}
    Determines whether or not to display warning text if the external glossary file hasn’t been generated due to an incomplete build.

  \item \texttt{postdot} \hspace{1cm} \textsection{glossaries-extra v1.12+} \hspace{1cm} \textsection{2.2; 12}
    A shortcut for \texttt{nopostdot=false}.

  \item \texttt{postpunc=} \texttt{(value)} \hspace{1cm} \textsection{glossaries-extra v1.21+} \hspace{1cm} \textsection{2.2; 12}
    An alternative to \texttt{postdot}, this can be used to insert a different punctuation character after the description.

    \begin{itemize}
      \item \texttt{postpunc=comma} \hspace{1cm} \textsection{13}
        Inserts a comma after the description.

      \item \texttt{postpunc=dot} \hspace{1cm} \textsection{13}
        Equivalent to \texttt{postdot} or \texttt{nopostdot=false}.

      \item \texttt{postpunc=none} \hspace{1cm} \textsection{13}
        Switches off automatic post-description punctuation insertion.

      \item \texttt{postpunc=} \texttt{(punctuation)} \hspace{1cm} \textsection{13}
        Inserts \texttt{(punctuation)} after the description.
    \end{itemize}

  \item \texttt{prefix} \hspace{1cm} \textsection{2.3; 14}
\end{itemize}
Package Option Summary

Loads glossaries-prefix.

**record**=⟨value⟩  
*default:* only; *initial:* off

Indicates whether or not bib2gls is being used (in which case entry indexing is performed by adding bib2gls records in the aux file).

**record**=alsoindex  
*alias:* hybrid

**record**=hybrid  
Performs a mixture of bib2gls records in the aux file (to select entries from a bib file) and makeindex/xindy indexing in their associated files. The actual sorting and collation is performed by the indexing application, so sort=none and save-locations =false should be used in `\GlsXtrLoadResources` (because it's redundant to make bib2gls sort and collate as well). This setting should be used with `\makeglossaries` before `\GlsXtrLoadResources` and glossaries should be displayed with `\printglossary` (or `\printglossaries`). There's little point in using this setting unless you have a custom xindy module that you can't convert to an equivalent set of bib2gls options.

**record**=nameref  
Entry indexing is performed by adding bib2gls nameref records in the aux file. Glossaries should be displayed with the "unsrt" family of commands.

**record**=off  
Entry indexing is performed as per the base glossaries package, using either `\makeglossaries` or `\makenoidxglossaries`.

**record**=only  
Entry indexing is performed by adding bib2gls records in the aux file. Glossaries should be displayed with the "unsrt" family of commands.

**shortcuts**=⟨{value}⟩  
*initial:* none

Defines various shortcut commands (boolean only with just the base glossaries package).

**shortcuts**=abbr  
*alias:* abbreviations

**shortcuts**=abbreviations  
Implements the abbreviation shortcuts.

**shortcuts**=abother  
Implements `shortcuts=abbreviations` and `shortcuts=other`.

**shortcuts**=ac  
Implements the acronym shortcuts that are compatible with `\newabbreviation`.

**shortcuts**=acoother
Package Option Summary

Implements \texttt{shortcuts=ac} and \texttt{shortcuts=other}.

\texttt{shortcuts=acro} \hspace{1cm} \textit{alias}: acronyms
\texttt{shortcuts=acronyms}

Implements the acronym shortcuts. Don’t use this option unless you have reverted \texttt{\newacronym} back to the base \texttt{glossaries} package’s acronym mechanism.

\texttt{shortcuts=all}

Implements \texttt{shortcuts=ac}, \texttt{shortcuts=abbreviations}, \texttt{shortcuts=other}.

\texttt{shortcuts=false} \hspace{1cm} \textit{alias}: all
\texttt{shortcuts=false}

Don’t define any shortcut commands.

\texttt{shortcuts=other}

Implements the shortcuts \texttt{\newentry}, \texttt{\newsym} and \texttt{\newnum}.

\texttt{shortcuts=true} \hspace{1cm} \textit{alias}: all
\texttt{shortcuts=true}

\texttt{showtargets=⟨value⟩}

Implements \texttt{debug=showtargets}.

\texttt{showtargets=anoteleft}

Markers are placed on either side of the link/target with the annotation on the left in all modes.

\texttt{showtargets=annoteright}

Markers are placed on either side of the link/target with the annotation on the right in all modes.

\texttt{showtargets=innerleft}

A marker and annotation are placed to the left of the link/target in all modes.

\texttt{showtargets=innerright}

A marker and annotation are placed to the left of the link/target in all modes.

\texttt{showtargets=left}

A marker is placed to the left of the link/target and a marginal note is used in outer mode.

\texttt{showtargets=right}

A marker is placed to the right of the link/target and a marginal note is used in outer mode.

\texttt{stylemods=⟨value⟩} \hspace{1cm} \textit{default}: default
\texttt{stylemods=⟨value⟩}

\texttt{§2.2; 13}

945
Package Option Summary

Loads glossaries-extra-stylemods with the given options. If `stylemods=default` then no options are passed to glossaries-extra-stylemods.

\begin{itemize}
  \item \texttt{stylemods=all}\hfill 13
  \texttt{Load glossaries-extra-stylemods with all predefined styles.}
  \item \texttt{stylemods=default}\hfill 13
  \texttt{Load glossaries-extra-stylemods with no options (patches any predefined styles that have been loaded).}
  \item \texttt{stylemods={list}}\hfill 14
  \texttt{Load glossaries-extra-stylemods with all the listed styles.}
\end{itemize}

\texttt{undefaction=⟨value⟩}\hfill $§2.4; 15$

Indicates whether to trigger an error or warning if an unknown entry label is referenced.

\begin{itemize}
  \item \texttt{undefaction=error}\hfill 15
  Trigger an error if an unknown entry label is referenced.
  \item \texttt{undefaction=warn}\hfill 15
  Trigger a warning if an unknown entry label is referenced.
\end{itemize}

\begin{quote}
\texttt{\usepackage[(options)]{glossaries}}\hfill \texttt{(automatically loaded with}\hfill 946
\texttt{\usepackage{glossaries-extra})}
\end{quote}

Base package. When loaded implicitly with glossaries-extra, any relevant options will be passed to the base glossaries package.

\begin{itemize}
  \item \texttt{acronym=⟨boolean⟩}\hfill $§2.1; 11$
  \texttt{If true, provides a new glossary with the label acronym and title given by \texttt{\acronymname}, redefines \texttt{\acronymtype} to acronym, and provides \texttt{\printacronyms}.}
  \item \texttt{acronymlists={⟨label-list⟩}}\hfill $♯\texttt{Ω}$
  \texttt{Identifies the glossaries that contain acronyms (defined with the base glossaries packages acronym mechanism).}
  \item \texttt{acronyms}\hfill $♯\texttt{Ω}$
  \texttt{Provides a new glossary with the label acronym, redefines \texttt{\acronymtype} to acronym, and provides \texttt{\printacronyms}.}
  \item \texttt{automake=⟨value⟩}\hfill \texttt{default: true; initial: false $♯\texttt{Ω}$}
  \texttt{Indicates whether or not to attempt to use \TeX’s shell escape to run an indexing application.}
\end{itemize}
Package Option Summary

\texttt{counter}=\langle\text{counter-name}\rangle \quad \text{\textit{initial: page}}

Sets the default location counter.

\texttt{debug}=\langle\text{value}\rangle \quad \text{\textit{initial: false}} \quad \text{\texttt{glossaries v4.24+}}

(modified by \texttt{glossaries-extra}'s \texttt{debug} option)

Adds markers to the document for debugging purposes.

\texttt{entrycounter}

Enables the entry counter for top-level entries.

\texttt{hyperfirst}=\langle\text{boolean}\rangle \quad \text{\textit{default: true; initial: true}}

Indicates whether or not to use hyperlinks on first use (if hyperlinks are supported).

\texttt{index}

Provides a new glossary with the label \texttt{index} and the title \texttt{\indexname}, and provides \texttt{\printindex} and \texttt{\newterm}.

\texttt{indexonlyfirst}=\langle\text{boolean}\rangle \quad \text{\textit{default: true; initial: false}}

Indicates whether or not to only index the first use.

\texttt{makeindex}

Indicates that the indexing should be performed by \texttt{makeindex} (default).

\texttt{mfirstuc} \quad \texttt{\texttt{glossaries v4.50+}}

Implements the expanded and unexpanded options provided by \texttt{mfirstuc}.

\texttt{nogroupskip}=\langle\text{boolean}\rangle \quad \text{\textit{default: true; initial: false}}

If true, suppress the gap between letter groups in the glossaries by default.

\texttt{nolist}

Don’t load \texttt{glossary-list}, which is normally loaded automatically. Note that if glossaries is loaded before \texttt{glossaries-extra}, then this option should be passed directly to glossaries not \texttt{glossaries-extra} otherwise it will be too late to implement.

\texttt{nolong}

Don’t load \texttt{glossary-long}, which is normally loaded automatically. Note that if glossaries is loaded before \texttt{glossaries-extra}, then this option should be passed directly to glossaries not \texttt{glossaries-extra} otherwise it will be too late to implement.

\texttt{nomain}

Prevents the definition of the \texttt{main} glossary. You will need to define another glossary to use instead. For example, with the \texttt{acronyms} package option.

\texttt{nonumberlist}

Set no location lists as the default for all glossaries. May be overridden for individual glossaries with \texttt{nonumberlist}=true.
nopostdot=⟨boolean⟩  
(default: true; initial: true)

If true, suppresses the automatic insertion of a full stop after each entry’s description in the glossary (for styles that support this). The default is nopostdot=true for glossaries-extra and nopostdot=false for just glossaries.

noredefwarn

Suppresses a warning if theglossary or \printglossary have already been defined (which indicates that the document class or another package also provides a mechanism for creating a glossary that could potentially conflict with glossaries). This option is automatically implemented with glossaries-extra.

nostyles

Don’t load the default set of predefined styles. Note that if glossaries is loaded before glossaries-extra, then this option should be passed directly to glossaries not glossaries-extra otherwise it will be too late to implement.

nosuper

Don’t load glossary-super, which is normally loaded automatically. Note that if glossaries is loaded before glossaries-extra, then this option should be passed directly to glossaries not glossaries-extra otherwise it will be too late to implement.

notranslate

Equivalent to translate=false.

notree

Don’t load glossary-tree, which is normally loaded automatically. Note that if glossaries is loaded before glossaries-extra, then this option should be passed directly to glossaries not glossaries-extra otherwise it will be too late to implement.

nowarn

Suppresses warnings.

numberedsection=⟨value⟩  
(default: nolabel; initial: false)

Indicates whether or not glossary section headers will be numbered and also if they should automatically be labelled.

numbers

Provides a new glossary with the label numbers and the title \glsnumbersgroupname, and provides \printnumbers. With glossaries-extra, this additionally defines \glsxtrnewnumber.

sanitizesort=⟨boolean⟩  
(default: true; initial: varies)

Indicates whether the default sort value should be sanitized (only applicable with sort=standard).

savenumberlist=⟨boolean⟩  
(default: true; initial: false)

If true, patches the location list encapsulator to save the location list. With bib2gls, use the save-locations resource option.
**Package Option Summary**

`section=⟨value⟩`  
*default: section*  
Indicates which section heading command to use for the glossary. The value may be one of the standard sectioning command’s control sequence name (without the leading backslash), such as `chapter` or `section`.

`seenoindex=⟨value⟩`  
*initial: error*  
Indicates what to do if the `see` key is used before the associated indexing files have been opened by \makeglossaries.

- `seenoindex=error`  
  Triggers an error if the `see` key is used before \makeglossaries.

- `seenoindex=ignore`  
  Does nothing if the `see` key is used before \makeglossaries.

- `seenoindex=warn`  
  Triggers a warning if the `see` key is used before \makeglossaries.

`sort=⟨value⟩`  
*initial: standard*  
Indicates how the `sort` key should automatically be assigned if not explicitly provided (for \makeglossaries and \makenoidxglossaries only).

- `sort=clear`  
  Sets the `sort` key to an empty value. Use this option if no indexing is required for a slightly faster build.

- `sort=def`  
  Use the (zero-padded) order of definition as the default for the `sort` key.

- `sort=none`  
  Don’t process the `sort` key. Use this option if no indexing is required for a slightly faster build.

- `sort=standard`  
  Use the value of the `name` key as the default for the `sort` key and implement the \glsprestandardsort hook.

- `sort=use`  
  Use the (zero-padded) order of use as the default for the `sort` key.

`style=⟨style-name⟩`  
Sets the default glossary style to `⟨style-name⟩`.

`subentrycounter`  
Enables the entry counter for level 1 entries.
Package Option Summary

**symbols**
Provides a new glossary with the label `symbols` and the title `\glssymbolsgrouplename`, and provides `\printsymbols`. With `glossaries-extra`, this additionally defines `\glsextrnewsymbol`.

**toc**=\langle boolean\rangle
*default: true; initial: true*
If true, each glossary will be automatically added to the table of contents. The default is `toc=false` with `glossaries` and `toc=true` with `glossaries-extra`.

**translate**=\langle value\rangle
*initial: babel*
Indicates how multilingual support should be provided, if applicable.

- **translate=babel**
  Uses babel's language hooks to implement multilingual support (default for `glossaries-extra` if babel has been detected).

- **translate=false**
  Don’t implement multilingual support (default if no language package has been detected).

- **translate=true**
  Uses translator’s language hooks to implement multilingual support (default for `glossaries` if a language package has been detected).

**writeglslabelnames**
Creates a file called `\jobname.glslabels` that contains all defined entry labels and names (for the benefit of autocompletion tools).

**writeglslabels**
Creates a file called `\jobname.glslabels` that contains all defined entry labels (for the benefit of autocompletion tools).

**xindy**=\langle\langle options\rangle\rangle
Indicates that the indexing should be performed by xindy.
Index

Symbols

?? (unknown entry marker) 15, 28, 37, 38, 511, 512, 703, 883
. (full stop or period) see full stop (.)
~ (non-breaking space) see non-breaking space (~)
\$ 535, 698
* (modifier) see \GlsXtrSetStarModifier
# 535, 728
+ (modifier) see \GlsXtrSetPlusModifier

A

\AB §4.3.2; Table 4.1; 653
\Ab §4.3.2; Table 4.1; 653

\ab §4.3.2; Table 4.1; 527, 653
abbreviation styles 628
footnote-desc see short-footnote-desc
footnote-em see short-em-footnote
footnote-sc see short-sc-footnote
footnote-sm see short-sm-footnote
footnote see short-footnote
long-desc-em see long-noshort-em-desc
long-desc-sc see long-noshort-sc-desc
long-desc-sm see long-noshort-sm-desc
long-desc see long-noshort-desc
long-em-noshort-em-desc-noreg 131, 629
long-em-noshort-em-desc 80, 131, 629
long-em-noshort-em-noreg 64, 131, 629
long-em-noshort-em 64, 79, 131, 629
long-em-short-em-desc 82, 87, 629
long-em-short-em 63, 81, 86, 132, 630
long-em see long-noshort-em
long-hyphen-noshort-desc-noreg 107, 149, 630, 833
long-hyphen-noshort-noreg 64, 107, 149, 630, 833
long-hyphen-postshort-hyphen-desc 106, 630
long-hyphen-short-hyphen-desc 105, 106, 630
long-hyphen-short-hyphen 103–105, 148, 149, 496, 630, 631, 833, 834
long-noshort-desc-noreg 107, 131, 631
Index

long-noshort-desc  Table 4.2; 73, 74, 76–78, 80, 131, 147, 168, 629, 629, 630, 631, 632, 834

long-noshort-em-desc  73, 78, 628, 631

long-noshort-em  78, 630, 631

long-noshort-noreg  107, 131, 631

long-noshort-sc-desc  73, 76, 628, 631

long-noshort-sc  63, 75, 632, 633

long-noshort-sm-desc  77, 77, 629, 632

long-noshort-sm  76, 632, 635

long-noshort  Table 4.2; 47, 74–76, 78, 79, 131, 147, 629, 631, 632, 635, 834

long-only-short-only-desc  112, 113, 155, 156, 632, 850

long-only-short-only  41, 111, 112, 154, 632, 633, 850

long-only-short-sc-only-desc  113, 633, 861

long-only-short-sc-only  112, 633, 678, 862

long-postshort-sc-user-desc  92, 138, 633, 837

long-postshort-sc-user  91, 92, 138, 334, 633, 837, 862

long-postshort-user-desc  90, 633

long-postshort-user  90, 91, 142, 238, 633, 857

long-sc  see long-noshort-sc

long-short-desc  Table 4.2; 82–84, 86, 87, 133, 134, 629, 633, 634, 835

long-short-em-desc  82, 86, 634

long-short-em  63, 81, 85, 634

long-short-sc-desc  Table 4.2; 82, 83, 634

long-short-sc  Table 4.2; 41, 42, 44, 81, 82, 171, 172, 634

long-short-sm-desc  Table 4.2; 82, 84, 634

long-short-sm  Table 4.2; 81, 84, 634

long-short-user-desc  89, 91, 138, 635, 837

long-short-user  64, 88–90, 135, 139–142, 633, 635, 774, 888–891

long-short  Table 4.2; 3, 42, 44, 50, 60, 67, 73, 81, 82, 84–86, 88, 103, 132, 133, 165, 171, 240, 628, 630, 631, 634, 635, 836

long-sm  see long-noshort-sm

long-noshort-em  see long-noshort-em

nolong-short-em  67, 73, 635

nolong-short-noreg  130, 635

nolong-short-sc  67, 69, 636

nolong-short-sm  67, 71, 636

nolong-short  67, 69, 71, 73, 130, 635, 636

postfootnote-desc  see short-postfootnote-desc

postfootnote-em  see short-em-postfootnote

postfootnote-sc  see short-sc-postfootnote

postfootnote-sm  see short-sm-postfootnote

postfootnote  see short-postfootnote

short-desc  see short-nolong-desc

short-em-desc  see short-em

short-em-nolong-desc  64, 127, 637

short-em-footnote  64, 114, 126, 628, 637

short-em-long-desc  94, 97, 637

short-em-long-em-desc  94, 99, 637

short-em-long-em  93, 98, 637

short-em-long  93, 97, 637

short-em-nolong-desc  66, 72, 73, 637, 638

short-em-nolong  65, 71, 72, 638

short-em-postfootnote-desc  129, 638

short-em-postfootnote  128, 636, 638

short-em  see short-em-nolong

short-footnote-desc  Table 4.2; 115–117, 119, 123, 127, 144, 145, 628, 637, 638, 641, 643, 804

short-footnote  Table 4.2; 50, 64, 114–116, 118, 122, 126, 628, 637, 638, 641, 643, 780, 804

short-hyphen-long-hyphen-desc  110, 639

short-hyphen-long-hyphen  108–110, 149, 152, 639, 817, 869

short-hyphen-postlong-hyphen-desc
Index

<table>
<thead>
<tr>
<th>Page</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>110, 639</td>
<td>short-hyphen-postlong-hyphen</td>
</tr>
<tr>
<td>628, 643</td>
<td>short-sm-long-desc</td>
</tr>
<tr>
<td>628, 643</td>
<td>short-sm-long</td>
</tr>
<tr>
<td>628, 643</td>
<td>short-sm-nolong-desc</td>
</tr>
<tr>
<td>628, 643</td>
<td>short-sm-nolong</td>
</tr>
<tr>
<td>888, 891, 892</td>
<td>short-sm-postfootnote-desc</td>
</tr>
<tr>
<td>124, 636, 644</td>
<td>short-sm-postfootnote</td>
</tr>
<tr>
<td>653, 941</td>
<td>short-sm-nolong</td>
</tr>
<tr>
<td>953</td>
<td>abbreviationsname</td>
</tr>
<tr>
<td>657</td>
<td>accsupp package</td>
</tr>
<tr>
<td>476, 653, 658</td>
<td>ABP</td>
</tr>
<tr>
<td>657</td>
<td>Abp</td>
</tr>
<tr>
<td>657</td>
<td>AC</td>
</tr>
<tr>
<td>657</td>
<td>Ac</td>
</tr>
<tr>
<td>657</td>
<td>AcF</td>
</tr>
<tr>
<td>657</td>
<td>Acfp</td>
</tr>
<tr>
<td>657</td>
<td>ac</td>
</tr>
<tr>
<td>657</td>
<td>acfp</td>
</tr>
<tr>
<td>657</td>
<td>acl</td>
</tr>
<tr>
<td>657</td>
<td>aclp</td>
</tr>
<tr>
<td>657</td>
<td>aclp</td>
</tr>
<tr>
<td>657</td>
<td>acp</td>
</tr>
<tr>
<td>657</td>
<td>Acr</td>
</tr>
<tr>
<td>657</td>
<td>Acrf</td>
</tr>
<tr>
<td>657</td>
<td>Acrfull</td>
</tr>
<tr>
<td>657</td>
<td>Acrfullpl</td>
</tr>
<tr>
<td>657</td>
<td>Acrlong</td>
</tr>
<tr>
<td>657</td>
<td>Acrlongpl</td>
</tr>
<tr>
<td>657</td>
<td>acrfull</td>
</tr>
<tr>
<td>657</td>
<td>acrfullpl</td>
</tr>
<tr>
<td>657</td>
<td>acrlong</td>
</tr>
<tr>
<td>657</td>
<td>acrlongpl</td>
</tr>
<tr>
<td>657</td>
<td>acrlong</td>
</tr>
<tr>
<td>657</td>
<td>acrlongpl</td>
</tr>
<tr>
<td>657</td>
<td>acrfull</td>
</tr>
<tr>
<td>657</td>
<td>acrfullpl</td>
</tr>
<tr>
<td>657</td>
<td>acrlong</td>
</tr>
<tr>
<td>657</td>
<td>acrlongpl</td>
</tr>
<tr>
<td>657</td>
<td>acrlong</td>
</tr>
<tr>
<td>657</td>
<td>acrlongpl</td>
</tr>
<tr>
<td>657</td>
<td>acrfull</td>
</tr>
<tr>
<td>657</td>
<td>acrfullpl</td>
</tr>
<tr>
<td>657</td>
<td>acrlong</td>
</tr>
<tr>
<td>657</td>
<td>acrlongpl</td>
</tr>
<tr>
<td>657</td>
<td>acrlong</td>
</tr>
<tr>
<td>657</td>
<td>acrlongpl</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Entry</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>short-sm-long</td>
<td>Table 4.2; 94, 96, 643</td>
</tr>
<tr>
<td>short-sm-nolong-desc</td>
<td>66, 70, 71, 643</td>
</tr>
<tr>
<td>short-sm-nolong</td>
<td>65, 69, 70, 643, 644</td>
</tr>
<tr>
<td>short-sm-postfootnote-desc</td>
<td>125, 644</td>
</tr>
<tr>
<td>short-sm-postfootnote</td>
<td>124, 636, 644</td>
</tr>
<tr>
<td>short-sm-nolong</td>
<td>see short-sm-nolong</td>
</tr>
<tr>
<td>abbreviationsname</td>
<td>§2.1; 9, 10, 592,</td>
</tr>
<tr>
<td>accsupp package</td>
<td>689</td>
</tr>
<tr>
<td>ABP</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>Abp</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>AC</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>Ac</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>AcF</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>Acfp</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>ac</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>acfp</td>
<td>§4.3.2; Table 4.1; 654</td>
</tr>
<tr>
<td>acl</td>
<td>§4.3.2; Table 4.1; 19, 656</td>
</tr>
<tr>
<td>aclp</td>
<td>§4.3.2; Table 4.1; 656</td>
</tr>
<tr>
<td>aclp</td>
<td>§4.3.2; Table 4.1; 656</td>
</tr>
<tr>
<td>acp</td>
<td>§4.3.2; Table 4.1; 656</td>
</tr>
<tr>
<td>Acr</td>
<td>§4.3.2; Table 4.1; 657</td>
</tr>
<tr>
<td>Acrf</td>
<td>§4.3.2; Table 4.1; 657</td>
</tr>
<tr>
<td>acrfull</td>
<td>655, 657</td>
</tr>
<tr>
<td>acrfullpl</td>
<td>655, 657</td>
</tr>
<tr>
<td>acrlong</td>
<td>656, 657</td>
</tr>
<tr>
<td>acrlongpl</td>
<td>656, 657</td>
</tr>
<tr>
<td>acrlong</td>
<td>656, 658</td>
</tr>
<tr>
<td>acrlongpl</td>
<td>656, 658</td>
</tr>
</tbody>
</table>
Index

acronym (base) style                       \afp                           §4.3.2; Table 4.1; 661
  dua-desc                                \AL                            §4.3.2; Table 4.1; 661
  dua                                      \A1                            §4.3.2; Table 4.1; 661
  footnote-desc                            \al                            §4.3.2; Table 4.1; 661
  footnote-sc-desc                         \align environment             26
  footnote-sc                               all caps 200, 253, 698, 846, 847, 905, 906
  footnote-sm-desc                         \ALP                           §4.3.2; Table 4.1; 661
  footnote-sm                              \Alp                           §4.3.2; Table 4.1; 661
  footnote                                  \alp                           §4.3.2; Table 4.1; 661
  long-sc-short-desc                       \Alph                          262, 263
  long-sc-short                             \alph                          262
  long-short 229                           \Alpha                         §11.5.8; 541, 552, 553, 573, 662
  long-sm-short-desc                       \alpha                         541, 551–553
  long-sm-short                             \alsoname 662, 933
  long-sp-short-desc                       \langle alt-mod \rangle (modifier) see \GlsXtrSetAltModifier
  long-sp-short                             \asmath package 27
  sc-short-long-desc                       \andname 662
  sc-short-long                             \app 164, 475
  short-long                                \apptoglossarypreamble §8.2; 377, 662
  sm-short-long-desc                       \arabic 262, 263
  sm-short-long                             \arara 511
  \acrpluralsuffix ☞ 36, 658
  \Acrshort ☞ 659
  \acrsuffix ☞ 18, 19, 50, 659
  \acrsuffixpl ☞ 659, 660
  \acrsuffixpl ☞ 659, 660
  \acs §4.3.2; Table 4.1; 659
  \acs §4.3.2; Table 4.1; 659
  \acs §4.3.2; Table 4.1; 19, 659
  \ACSP §4.3.2; Table 4.1; 660
  \acsp §4.3.2; Table 4.1; 660
  \acsp §4.3.2; Table 4.1; 660
  \actualchar 576
  \AF §4.3.2; Table 4.1; 660
  \Af §4.3.2; Table 4.1; 660
  \af §4.3.2; Table 4.1; 660
  \AFP §4.3.2; Table 4.1; 660
  \Afp §4.3.2; Table 4.1; 661

B

babel package 3, 4, 9, 17, 492, 592, 653, 662, 950
  \backmatter 188
  \beamer class 286
  \Beta §11.5.8; 541, 552, 573, 663
  \beta 541, 551, 553
  \bib2gls 508–573
  \--g see --group
  \--group 24, 26, 34, 273, 390, 395, 397, 426, 442, 448, 463, 519, 537, 607
  \--i see --index-conversion

954
Index

case change ............................ 366, 499, 567, 899, see also
capital case, lowercase, title case, sentence case & all caps
categories
  abbreviation §10.1; 41, 42, 58–60, 159,
  238, 240, 320, 422, 493, 628, 852,
  916, 933
  acronym §10.1; 5, 42, 46, 58–60, 181,
  236, 238, 240, 422, 493, 495, 628,
  852, 917
general §10.1; 238, 245, 253, 311, 312,
  316, 422, 468, 493, 495, 503, 505, 852
  index §10.1; 12, 493, 852
  number §10.1; 11, 494, 853
  symbol §10.1; 10, 402, 494, 853
  term .................................... 422
category attributes ...................... 247, 608, 610, 796
accessaposplural ....................... §9.1; 476
accessinsertdots ...................... §9.1; 476
accessnoshortplural ................. §9.1; 476
aposplural §10.2.1.1; 43, 162, 476, 497,
  498, 754
dualindex §10.2.1.4; 502, 574, 576, 586
encapinnerfmt .......................... 243
encapnocaseinnerfmt ................. 243
ecount .................................. §10.2.1.4; 308, 309,
  311–313, 501
eexternallocation §10.2.1.4; 195, 196,
  503, 555, 876
firstshortaccess §9.1; 168, 169, 477,
  506, 607
glossdesc §10.2.1.3; 197, 200, 234, 418,
  499, 501
glossdescfont §10.2.1.3; 418, 500, 501
glossname §10.2.1.3; 197, 418, 419,
  458, 500
glossnamefont §10.2.1.3; 415, 418,
  419, 458, 501, 751
glosssymbolfont §10.2.1.3; 418, 501
headuc §10.2.1.4; 204, 216–218, 501,
  812–816, 820
hyperoutside §10.2.1.2; 189, 499
indexname §10.2.1.3; 420, 501, 574, 586
indexonlyfirst §10.2.1.2; 265,
  499, 576
innertextformat ....................... 243, 784
insertdotformat §10.2.1.1; 162, 476, 497
linkcount §10.2.1.4; 317, 320, 501

booktabs package ..................... 470, 647

C
\capitalisefmtwords ............... 200, 500, 664,
  698, 937
\capitalisewords 200, 291, 373, 500, 665,
  698, 802
\caption .................................. 27

955
Index

linkcountmaster . §10.2.1.4; 317, 502
markshortwords §10.2.1.1; 43, 63, 103, 148, 162, 167, 168, 174, 177, 497, 631, 639, 721, 778, 893
markwords . §10.2.1.1; 43, 63, 103, 148, 162, 168, 176, 496, 497, 630, 631, 639, 721, 779, 893
multioptions . §7.8; 332, 353, 354
nameshortaccess . §9.1; 168, 169, 476, 605
nohyper . §10.2.1.2; 256, 499, 505
nohyperfirst §10.2.1.2; 115, 116, 186, 189, 256, 498, 639, 792
nohypernext . §10.2.1.2; 186, 256, 498
noshortplural §10.2.1.1; 44, 162, 476, 497, 498, 653, 754
pluraldiscardperiod §10.2.1.1; 495
recordcount . 524–526, 822, 867
regular . §10.2.1.1; 494
retainfirstuseperiod §10.2.1.1; 247, 330, 495, 796

tagging . §10.2.1.1; 58, 59, 498
targetcategory . §10.2.1.4; 503
targetname . §10.2.1.4; 502, 503
targeturl . §10.2.1.4; 36, 256, 502, 588
textformat . §10.2.1.2; 191, 238, 239, 499
textshortaccess . §9.1; 168, 169, 477, 608, 611
unitcount . 313
wrgloss §10.2.1.2; 185, 189, 194, 264, 499
category post-description hook . 10–12, 421, 422, 428, 599, 603, 853
category post-link hook . 235, 237, 245–248, 250, 251, 423, 563, 599, 603, 855
category post-name hook . 563, 599
\cGLS §6.1; Table 4.1; 311, 653, 654, 665, 666, see also \glsenableentrycount & \cGLSformat
\cGLS . §6.1; 310, 653, 654, 665, 666, see also \glsenableentrycount & \cGLSformat
\cglis §6.1; Table 4.1; 56, 183, 308, 309, 313, 501, 524, 527, 653, 654, 665, 666, see also \glsenableentrycount & \cGLSformat
\cGLSformat . §6.1; 311, 665
\cGLSformat . §6.1; 310, 666
\cGLSformat . §6.1; 309–311, 524, 525, 666
\cGlspl . §6.1; Table 4.1; 311, 654, 656, 666, see also \glsenableentrycount & \cGlsplformat
\cGlspl . §6.1; 310, 654, 656, 666, 667, see also \glsenableentrycount & \cGlsplformat
\cGLSformat . §6.1; Table 4.1; 310, 654, 657, 666, 667, see also \glsenableentrycount & \cGLSformat
\cGlsformat . §6.1; Table 4.1; 311, 665
\cGLSformat . §6.1; 311, 666
\cGLSformat . §6.1; 310, 667
\chapter . 431, 675
chapter counter . 313, 529, 530, 533, 555
\char . 385
\Chi . §11.5.8; 573, 667
CLI . 599, 601
convertgls2bib . 272, 508
\--index-conversion . 272
\csdef . 38, 39, 829
\cGLSxtrLetField . §3.5; 39, 40, 667
\cslet . 40, 667, 829
\csletcs . 40, 667
\currentglossary . 399, 414, 431, 667
\CurrentTrackedDialect . 593
\CurrentTrackedLanguage . 593
\CustomAbbreviationFields . §4.5.3.1; 165, 167, 168, 667

D

datatool–base package . 302, 303, 671, 801, 819, 822
datatool package . 304, 507, 671, 730, 801
\defglsentryfmt . 81, 185, 254, 667, 698, 700, 731–733
\delimN . 668, 729, 752
Index

\ delimR ........................................ 668, 729, 752
\ Delta ........................................ 552
description environment .................. 646, 940
descriptionname ......................... 446, 469, 668
\dGLS .................................................. §11.5.7; 565, 668, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\dGl .................................................. §11.5.7; 565, 668, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\dgls .................................................. §11.5.7; 183, 564, 569, 570, 668, 669, 670, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\dGlsdisp ........................................... §11.5.7; 565, 669, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\dGlsfield .......................................... §11.5.7; 566, 567, 669, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\dGlslink .......................................... §11.5.7; 565, 670, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\dGLSlink .......................................... §11.5.7; 565, 670, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\dGLSpl ........................................... §11.5.7; 565, 670, see also
\glsxtraddlabelprefix & \glsxtrprependlabelprefix
\Digamma .......................................... §11.5.8; 551, 552, 573, 671
display full form. 50, 63, 81, 88, 93, 100, 103, 104, 106, 109–111, 132, 170, 172, 173, 175, 599, 601, 806, 807
document environment . 16, 17, 37, 512, 578, 942, 943
\dolifglossarynoexistsordo 671, see also
\ifglossaryexists & \glsxtrifemptyglossary
\dolistcsloop ........................................ 304, 800
\DTLformatlist .................................... 302, 304, 671, 801
\DTLifinlist ........................................ 303, 507, 671, 730, 819, 822

\eglssetwidest ................................... §8.6.5.4; 434, 671, 937
\eglsupdatewidest ................................. §8.6.5.4; 434, 672, 938
\eGlsXtrSetField ................................... §3.5; 39, 40, 672
\emph ............................................. 156, 158, 629–631, 634, 635, 637, 638
\encapchar ......................................... 576
entry location .................................... 599, 601, 602
\entryname ........................................ 446, 469, 672
\Epsilon ........................................... §11.5.8; 573, 672
equation counter ................. 26, 27, 555, 556, 558, 800, 943
equation environment .............. 26, 27
\Eta ................................................ §11.5.8; 573, 672
etoolbox package . 38–40, 304, 305, 318, 407, 505, 513, 614, 615, 667, 728, 800–802, 819, 829, 894, 895
\example-glossaries-acronym.bib ........................................... §14.1; 581
\example-glossaries-acronym-desc.bib ................................... §14.1; 581
\example-glossaries-abbreviations.bib .................................. §14.1; 581
\example-glossaries-abbreviations-lang.bib ................................ §14.1; 581
## Index

<table>
<thead>
<tr>
<th>example-glossaries</th>
<th>§14.1; 581</th>
</tr>
</thead>
<tbody>
<tr>
<td>-childmultipar.bib</td>
<td>§14.1; 581</td>
</tr>
<tr>
<td>example-glossaries-childnoname.bib</td>
<td>§14.1; 581</td>
</tr>
<tr>
<td>example-glossaries-cite.bib</td>
<td>§14.1; 581</td>
</tr>
<tr>
<td>example-glossaries-images.bib</td>
<td>§14.1; 581</td>
</tr>
<tr>
<td>example-glossaries-long.bib</td>
<td>§14.1; 581</td>
</tr>
<tr>
<td>example-glossaries-longchild.bib</td>
<td>§14.1; 582</td>
</tr>
<tr>
<td>example-glossaries-multipar.bib</td>
<td>§14.1; 582</td>
</tr>
<tr>
<td>example-glossaries-parent.bib</td>
<td>§14.1; 582</td>
</tr>
<tr>
<td>example-glossaries-symbolnames.bib</td>
<td>§14.1; 582</td>
</tr>
<tr>
<td>example-glossaries-symbols.bib</td>
<td>§14.1; 582</td>
</tr>
<tr>
<td>example-glossaries-url.bib</td>
<td>§14.1; 582</td>
</tr>
<tr>
<td>example-glossaries-xr.bib</td>
<td>§14.1; 582</td>
</tr>
<tr>
<td>example-glossaries-xr.tex</td>
<td>§14.1; 581</td>
</tr>
<tr>
<td>\expandafter</td>
<td>166, 254</td>
</tr>
<tr>
<td>\expandonce</td>
<td>133, 134</td>
</tr>
<tr>
<td>\ExtraCustomAbbreviationFields</td>
<td>§4.5.3.1; 162, 164, 672</td>
</tr>
</tbody>
</table>

### F

- fancyhdr package ........................................ 444
- \fbox .................................................. 242, 243
- figure environment ........................................ 27
- file formats environment .................................. 27
- acr ..................................................... 6
- aux .................................................... 17, 26, 371, 407, 508, 513, 601, 614, 927, 942
- bib .................................................... 17, 33, 34, 508, 581, 602
- glo-abr ................................................. 9
- glo-br .................................................. 9
- gls ..................................................... 17
- glsdefs ............................................... 16, 17, 37
- glslabels ............................................. 950
- glstex ............................................... 34, 508, 511, 603
- ldf ................................................... 592, 593
- log .................................................... 942
- tex .................................................... 33, 508, 581
- toc .................................................... 215

### G

- gettitlestring package ................................... 215, 735
- \GetTitleStringDisableCommands ......................... 215
- \GetTitleStringSetup .................................. 215
- \glssetwidest ........................................... §8.6.5.4; 434, 673
- \glsupdatewidest ........................................ §8.6.5.4; 434, 674
- \glsXtrSetField ........................................ §3.5; 39, 40, 674

GUI ......................................................... 600
Index

Glo

\golinkprefix 196, 256, 379, 409, 502, 626, 674
glossaries--extra--stylemods package 13, 14, 383, 385, 393, 421, 427, 431, 463, 554, 589, 603, 671–674, 697, 714–716, 727, 731, 734, 735, 754, 769–774, 783, 858, 882, 883, 937, 938, 941, 946

all 941

⟨name⟩ 941
glossaries--extra package §2; 2, 8, 941
abbreviations §2.1; 9, 10, 14, 46, 534, 592, 658, 780, 926, 927, 941
accsupp §2.3; 14, 168, 169, 416, 475, 605–611, 679–681, 689–695, 700, 780, 781, 784, 858, 941
autoseeindex §2.4; 21, 22, 33, 263, 270, 276, 302, 941
false 22–24, 33, 609, 941
true 941
bibglsaux §2.4; 26, 663, 864, 942
debug §2.5; 28, 942, 947
all 929, 942
false 28, 942
showaccsupp 28, 942
showtargets 29, 30, 761, 762, 942, 945, see also showtargets
showwrgloss 29, 893, 942
true 28, 29, 942
dodec §2.4; 16, 324, 370, 371, 578, 942
atom 17, 942
false 16, 942
restricted 942
true 16, 37, 308, 376, 942
equations §2.4; 26, 555, 943
floats §2.4; 27, 943
indexcounter §2.4; 25, 27, 29, 558, 559, 573, 614, 942, 943
indexcrossrefs §2.4; 20, 22, 23, 265, 266, 279, 943
false 20, 21, 24, 943
true 20, 21, 943
nomissingglstext §2.1; 9, 943
postdot §2.2; 2, 12, 421, 422, 849, 943, see also nopostdot & postpunc
postpunc §2.2; 12, 13, 421, 422, 427, 754, 943
comma 13, 943
dot 13, 943
none 13, 943
⟨punctuation⟩ 943
prefix §2.3; 14, 54, 367, 943
record §2.4; 23, 944
alsoindex see hybrid
hybrid 23, 25, 29, 37, 510, 512, 533, 944
nameref 23, 24, 34, 196, 279, 372, 389, 510, 512, 533, 555, 601, 607, 779, 797, 944
off 23, 37, 385, 944
only 23, 24, 34, 37, 279, 372, 389, 512, 533, 607, 779, 944
shortcuts §2.4; 17–19, 308, 326, 944, 945
abbr see abbreviations
abbreviations 18, 19, 56, 653, 654, 660–663, 795, 916, 944
abother 19, 944
ac Table 4.1; 18, 19, 43, 56, 654–657, 659, 660, 795, 916, 944
acother 19, 795, 944
acro see acronyms
acronyms 19, 654–657, 659, 660, 945

959
Index

all ........................................ 19, 20, 795, 945
false ................................. see all
none .................................. 20, 945
other .............................. 18, 795, 917, 919, 945
true ................................. see all
showtargets §2.5; 29, 380, 872, 873, 945
annoteleft ............ 31, 338, 945
annoteright .......... 31, 945
innerleft .......... 31, 945
innerright .......... 31, 945
left .................. 30, 945
right ................. 30, 945
stylemods  §2.2; 12–14, 384, 385, 387, 389, 427, 437, 941, 945
all .................. 13, 946
default .................. 13, 14, 946
(list) .................. 14, 946
undefaction §2.4; 15, 16, 281, 283, 703, 822, 883, 897, 946
error .................. 15, 24, 703, 704, 818, 820, 946
warn .................. 15, 17, 24, 39, 277, 511, 703, 704, 714, 818, 820, 865, 946
glossaries–prefix package 14, 54, 55, 207–210, 218, 367, 590, 608, 609, 755, 915, 920–925, 944
glossaries package ................. 2, 41, 946
acronym §2.1; 10, 11, 534, 658, 941, 945
acronymlists ☐ .......... 10, 946
acronyms §2.1; 11, 44, 46, 534, 658, 926, 927, 941, 946, 947
automake .................. 596, 848, 946
counter ............ 27, 261, 529, 555, 605, 943, 947
debug .................. 947
entrycounter .......... 377, 414, 416, 446, 456, 706, 756, 947
hyperfirst .............. 186, 256, 498, 947
index  §2.1; 11, 12, 422, 535, 852, 919, 926, 929, 947
indexonlyfirst ...... 264, 265, 499, 576, 896, 947
makeindex ............ 556, 947
mfirstuc .................. 947
nogroupskip 428, 430, 461, 728, 896, 947
nolist .................. 14, 947
nolong .................. 947
nomain .................. 6, 701, 947
nonumberlist 261, 398, 423, 461, 529, 602, 675, 751, 947
nopostdot §2.2; 2–4, 12, 13, 428, 754, 943, 948, see also postdot & postpunc
noredefwarn .................. 3, 948
nostyle 14, 427, 433, 437, 948
nosuper .................. 948
notranslate ............ 948
notree .................. 433, 948
nowarn .................. 674, 675, 948
numberedsection .......... 378, 625, 675, 698, 948
numbers §2.1; 11, 19, 32, 422, 534, 752, 847, 852, 919, 927, 929, 948
sanitizesort ............ 948
savenumberlist ........ 514, 948
section .................. 442, 675, 949
seenoindex .................. 23, 949
error .................. 949
ignore .................. 949
warn .................. 949
sort .................. 389, 412, 949
clear .................. 949
def .................. 949
none .................. 24, 949
standard ............. 755, 948, 949
use .................. 949
style .................. 949
subentrycounter ...... 377, 414, 416, 446, 472, 756, 762, 949
symbols §2.1; 10, 18, 32, 422, 534, 561, 764, 847, 853, 919, 927, 929, 950
toc .................. 2–4, 950
translate .................. 3, 4, 948, 950
babel .................. 950
false .................. 950
true .................. 950
writeglslabelnames ...... 17, 950
writeglslabels ........ 17, 950
xindy ............. 261, 263, 426, 574, 950
Index

\GlossariesAbbrStyleTooComplex-
Warning .......................... 674
\glossariesextrasetup §2; 8, 14, 18, 534, 674, 934
\GlossariesExtraWarning 160, 674, 842, 848, 875, 883, 892, 915
\GlossariesExtraWarningNoLine ... 675
glossary-bookindex package §8.7.1; 14, 387, 437–444, 646, 788–791
glossary-inline package ........... 428, 646, 732, 754, 755
glossary-list package 645–647, 734, 735, 947
glossary-long package ....... 649, 702, 752, 947
glossary-longbooktabs package 444, 468, 647
glossary-longextra package §8.7.2; 14, 444–461, 553, 645, 647–649, 737–748, 896
glossary-longragged package .... 649
glossary-mcols package ........... 649, 650
glossary-super package .... 650, 702, 752, 948
glossary-superragged package .... 650
glossary-table package §8.7.4; 468–474, 650, 764, 930
glossary-topic package §8.7.3; 461–468, 650, 766–769
glossary-tree package 431, 433, 435, 443, 461, 463, 645, 646, 650, 651, 715, 760, 769–773, 948
glossary ......................... 600
glossary entry fields ................. 614
coldcount .......................... 554, 614, 820
coldlist ............................. 472, 614
coldindexcounter .................... 573, 614, 824
coldloclist 35, 397, 398, 513, 556, 614, 928
coldrecordcounter ................. 522, 523, 614, 882
coldrecordcounter.(counter) ....... 522, 523, 615, 859
coldrecordcounter.(counter).(location) 522, 523, 615, 831
coldrecord.(counter) ............... 407, 408, 615, 779
coldsecondarygroup 390, 391, 395, 396, 615
glossary entry keys .................. 33, 605
access ......................... 162, 475, 476, 605, 690, 751
alias §3.2; 20, 33, 36, 37, 263, 265, 267, 270, 272, 274, 276, 277, 279, 291, 325, 370, 581, 588, 605, 782, 783, 863, 887
category §3.2; 12, 33, 41, 42, 59, 60, 162, 265, 396, 402, 492, 493, 528, 605, 628, 726, 729, 847, 864, 916, 917
counter ........................... 261, 262, 605
description §12, 14, 44, 63, 64, 251, 258, 422, 446, 448, 480, 487, 516, 599, 605, 606, 631, 634, 638–641, 679, 702, 704, 705, 849, 856, 860, 895, 919
descriptionaccess ................. 606, 679, 702
descriptionplural §8.7.4; 480, 487, 606, 680, 702, 705
descriptionpluralaccess ........... 606, 680, 702
first §3.2; 36, 61, 163, 167–170, 204, 225, 226, 228, 255, 346, 478, 486, 575, 606, 635, 639, 640, 680, 705, 717, 721, 722, 781, 811, 812, 877, 878
firstaccess ....................... 162, 477, 606, 680, 719
firstplural §3.2; 36, 163, 167–169, 204, 226, 255, 479, 486, 606, 607, 635, 639, 640, 681, 705, 706, 720, 722, 781, 812, 878
firstpluralaccess ................. 162, 477, 606, 681, 720
group §3.2; 24, 34, 387–391, 395, 396, 426, 514, 519, 520, 602, 607, 928
location §3.2; 24, 34, 35, 397, 398, 428, 440, 469, 513, 514, 530, 556, 572, 607, 614, 703, 707, 928
long §3.2; 36, 43, 61, 175, 204, 233, 346, 364, 458, 469, 475, 481, 489, 495, 575, 607, 689, 706, 806, 831, 832, 896, 908
longaccess ........................ 162, 475, 607, 689, 736
longplural §3.2; 36, 43–45, 162, 176, 204, 475, 482, 489, 607, 608, 689, 707, 807, 834, 835
longpluralaccess ................. 162, 475, 608, 689, 749
name §3.2; 34, 36, 44, 133–135, 138, 139, 144, 146, 147, 155, 156, 163, 168–170, 204, 223, 224, 228, 237, 292, 369, 372, 389, 415, 419, 446, 448, 475, 477, 485, 553, 563, 574, 575, 579, 586, 589, 590, 599,
Index

<table>
<thead>
<tr>
<th>Page Range</th>
<th>Index Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>605, 608, 610, 611, 677, 690, 707, 724, 725, 751, 780, 800, 811, 814, 837, 847, 871, 874, 919, 949</td>
<td>parent</td>
</tr>
<tr>
<td>37, 325, 379, 387, 426, 435, 554, 575, 608, 707, 800, 875, 895, 896</td>
<td>plural</td>
</tr>
<tr>
<td>35–37, 163, 168, 169, 204, 224, 225, 255, 478, 485, 606, 608, 635, 639, 640, 690, 708, 725, 754, 780, 814, 815, 881</td>
<td>pluralaccess</td>
</tr>
<tr>
<td>616, 619, 92, 202, 204, 224, 225, 255, 478, 485, 606, 608, 635, 639, 640, 690, 708, 725, 754, 780, 814, 815, 881</td>
<td>prefix</td>
</tr>
<tr>
<td>56, 209, 608, 609, 921, 923</td>
<td>prefixfirst</td>
</tr>
<tr>
<td>56, 210, 609, 921, 924</td>
<td>prefixplural</td>
</tr>
<tr>
<td>55, 208, 609, 922, 925</td>
<td>see</td>
</tr>
<tr>
<td>§3.2; 20–22, 33, 263, 265–267, 274, 276–279, 291, 299, 325, 369, 441, 581, 609, 784, 863, 892, 941</td>
<td>short</td>
</tr>
<tr>
<td>36, 43, 61, 159, 177, 204, 255, 346, 364, 365, 369, 458, 475, 476, 481, 488, 495, 498, 609, 610, 691, 706, 709, 780, 806, 808, 868, 896, 905, 911</td>
<td>shortaccess</td>
</tr>
<tr>
<td>162, 164, 475–477, 609, 691, 701, 760, 784</td>
<td>shortplural</td>
</tr>
<tr>
<td>36, 43–45, 162, 178, 204, 475, 481, 488, 497, 498, 610, 653, 691, 709, 807, 871, 872</td>
<td>shortpluralaccess</td>
</tr>
<tr>
<td>616, 619, 620, 753, 761, 764</td>
<td>sort</td>
</tr>
<tr>
<td>10, 11, 17, 34, 37, 134, 145, 156, 229, 388, 389, 412, 497, 574, 575, 610, 847, 949</td>
<td>symbol</td>
</tr>
<tr>
<td>251, 365, 366, 419, 448, 452, 456, 479, 486, 563, 610, 611, 692, 709, 710, 763, 896, 912</td>
<td>symbolaccess</td>
</tr>
<tr>
<td>419, 610, 692, 710, 763</td>
<td>symbolplural</td>
</tr>
<tr>
<td>479, 487, 610, 611, 692, 710, 763</td>
<td>symbolpluralaccess</td>
</tr>
<tr>
<td>611, 692, 763</td>
<td>text</td>
</tr>
<tr>
<td>35–37, 61, 163, 168–170, 204, 224, 225, 255, 346, 369, 478, 485, 567, 606, 608, 611, 635, 639, 640, 693, 710, 726, 765, 780, 816, 882</td>
<td>textaccess</td>
</tr>
<tr>
<td>162, 477, 611, 693, 765</td>
<td>type</td>
</tr>
<tr>
<td>41, 44, 254, 262, 376, 387, 414, 426, 492, 536, 611, 668, 711, 847, 899, 917, 919</td>
<td>user1</td>
</tr>
<tr>
<td>63, 64, 135, 295, 328, 366, 367, 394, 395, 419, 482, 489, 495, 602, 611, 693, 711, 775</td>
<td>user1access</td>
</tr>
<tr>
<td>611, 693, 775</td>
<td>user2</td>
</tr>
<tr>
<td>293, 366, 367, 482, 490, 612, 693, 711, 775</td>
<td>user2access</td>
</tr>
<tr>
<td>612, 693, 775</td>
<td>user3</td>
</tr>
<tr>
<td>483, 490, 612, 694, 711, 712, 776</td>
<td>user3access</td>
</tr>
<tr>
<td>612, 694, 776</td>
<td>user4</td>
</tr>
<tr>
<td>483, 491, 612, 694, 712, 777</td>
<td>user4access</td>
</tr>
<tr>
<td>612, 694, 777</td>
<td>user5</td>
</tr>
<tr>
<td>484, 491, 612, 613, 695, 712, 777</td>
<td>user5access</td>
</tr>
<tr>
<td>613, 695, 777</td>
<td>user6</td>
</tr>
<tr>
<td>484, 491, 613, 695, 712, 713, 778</td>
<td>user6access</td>
</tr>
<tr>
<td>613, 695, 778</td>
<td>glossary, ignored</td>
</tr>
<tr>
<td>see ignored glossary</td>
<td></td>
</tr>
<tr>
<td>glossary, mini</td>
<td></td>
</tr>
<tr>
<td>see mini-glossary</td>
<td></td>
</tr>
<tr>
<td>glossary styles</td>
<td></td>
</tr>
<tr>
<td>645</td>
<td>abbr–long–short</td>
</tr>
<tr>
<td>§8.7.2.6; 460, 645, 740, 744, 745</td>
<td>abbr–short–long</td>
</tr>
<tr>
<td>§8.7.2.6; 460, 645, 740, 741, 744, 745</td>
<td>altlist</td>
</tr>
<tr>
<td>429, 645, 697</td>
<td>alttree</td>
</tr>
<tr>
<td>431–433, 435, 437, 447, 461, 463, 553, 554, 589, 645, 646, 697, 783</td>
<td>alttreegroup</td>
</tr>
<tr>
<td>645</td>
<td>bookindex</td>
</tr>
<tr>
<td>§8.7.1; 14, 389, 394, 437–444, 534, 646, 788–791</td>
<td>index</td>
</tr>
<tr>
<td>432, 646, 649, 769–773</td>
<td>indexgroup</td>
</tr>
<tr>
<td>646, 731, 770, 772</td>
<td>inline</td>
</tr>
<tr>
<td>428, 646</td>
<td>list</td>
</tr>
<tr>
<td>14, 428, 429, 645, 646, 647, 734, 735, 940</td>
<td>listdotted</td>
</tr>
<tr>
<td>429, 646, 734</td>
<td>listgroup</td>
</tr>
<tr>
<td>429, 646, 647, 735</td>
<td>listhypergroup</td>
</tr>
<tr>
<td>647</td>
<td>long–booktabs</td>
</tr>
<tr>
<td>647</td>
<td></td>
</tr>
</tbody>
</table>
Index

long-desc-name §8.7.1; 449, 647, 737
long-desc-sym-name §8.7.2; 452, 647, 738
long-desc-sym §8.7.2.5; 448, 457, 647, 737, 738
long-loc-desc-name §8.7.2.3; 453, 647, 739
long-loc-desc-sym-name §8.7.2.4; 455, 647, 739
long-loc-sym-desc-name §8.7.2.4; 455, 648, 740
long-name-desc-loc §8.7.2.3; 452, 648, 741
long-name-desc-sym §8.7.2.4; 454, 648, 742
long-name-desc-sym §8.7.2.2; 450, 648, 741, 742
long-name-desc §8.7.2.1; 448, 449, 458, 648, 741, 742
long-name-sym-desc-loc §8.7.2.4; 454, 648, 743
long-name-sym-desc §8.7.2.2; 448, 451, 456, 649, 743
long-sym-desc-name §8.7.2.2; 451, 649, 747
long-sym-desc §8.7.2.5; 448, 457, 649, 747
long §14, 399, 428, 446, 649
longragged 649
mcolindex 649
mcolindexgroup 437, 649
mcoltree 462, 650
super 650
superragged 650
table §8.7.4; 468, 469, 472, 650, 930
topic §8.7.3; 461, 462, 650
topicmcols §8.7.3; 461, 650, 766
tree 431, 432, 461, 465, 650, 651, 769–773, 882, 883
treecol 430, 650, 651, 770, 772
treecolgregation 430, 651, 771
treenoname 432, 651, 771, 772
glossaryentry counter 935
\glossaryentrynumbers 398, 423, 675, 805
\glossaryheader 399, 675
\glossaryname 675
\glossarypostamble 382, 625, 675
\glossarypreamble 377, 382, 383, 625, 675, see also \setglossarypreamble
\glossarysection 675
\glossarytitle 399, 676
\glossarytoctitle 399, 676
\glossentry 398–400, 414, 472, 676
\Glossentrydesc 159, 466, 676, 701
\glossentrydesc 159, 418, 446, 499, 500, 605, 676, 702
\Glossentrynameother §8.6; 159, 415, 419, 676, 856
\Glossentrysymbol 159, 677, 763
\Glossentrysymbol 159, 418, 449, 501, 677, 708, 763
\glossxtrsetpopts §5.4; 231, 677, 851

Gls

\GLS 677
\Gls 677
\gls 600
\gls-like and \glstext-like options 186–196, 616
counter 27, 261, 523, 529, 557, 558, 616, 918
format 616
hyper 183, 186, 189, 204, 205, 231, 256, 320, 321, 498, 499, 616, 792
hyperoutside §5.1.2; 186, 189, 190, 234, 256, 499, 616, 824
innertextformat §5.1.2; 43, 166, 174, 191, 601, 616, 794, 795
local 191, 283, 617
noindex §5.1.2; 37, 51, 185, 186, 188, 190, 193, 194, 204, 205, 231, 259, 265, 278, 279, 576, 617, 820

963
postunset . . §5.1.2; 186, 189–193, 283, 617, 732 681, 682 \glsaccessfirstplural . . §9.2; 479, 680, 681, 682
prefix . . . . . . §5.1.2; 196, 382, 617
prerset . . . . . . §5.1.2; 43, 47, 185, 186, 189, 191–193, 282, 283, 287, 361, 617, 717, 720, 732
preunset . . . . . . §5.1.2; 43, 47, 185, 186, 189, 192, 282, 283, 285, 286, 362, 617, 732, 753, 754, 765
 textformat . . . . §5.1.2; 190, 191, 239, 617 \GLSaccessfmtfirst . . . . §9.3; 486, 682
theHValue . . . . §5.1.2; 194, 196, 618 \GLSaccessfmtfirstplural . . §9.3; 486, 682
type . . . . . . . . 186, 618, 696 \GLSaccessfmtfirstplural . . §9.3; 486, 682
wrgloss . . . . . . §5.1.2; 185, 186, 189, 190, 194, 264, 499, 618, 825, 897
\glsabbrvdefaultfont . . 66, 74, 82, 93, 115, 132, 133, 136, 148, 154, 678
\glsabbrvemfont . . . . 71–73, 78–80, 85–87, 97–99, 126–129, 158, 678
\glsabbrvfont . . . . 167, 168, 171, 174, 658, 678, 732, 778
\glsabbrvhyphenfont . . . 64, 148, 678
\glsabbrvonlyfont . . . . 154, 678
\glsabbrvscfont . . . . 67–69, 75, 76, 82, 83, 94, 95, 118–121, 136, 155–157, 678
\glsabbrvsconlyfont . . . . 113, 154, 155, 678
\glsabbrvuserfont . . . . 91, 136, 137, 679
\glsabbrvsmfont . . . . 69–71, 76, 77, 84, 96, 122–125, 157, 158, 679
\glsabbrvuserfont . . . . 88, 100, 135, 136, 679
\glsabspace . . . . Table 4.2; 132, 679
\GLSaccessdesc . . . . §9.2; 480, 679, 681
\GLSaccessdesc . . . . §9.2; 480, 679, 681
\glsaccessdesc . . . . §9.2; 480, 679, 681
\GLSaccessdescpl . . . . §9.2; 480, 680, 681
\GLSaccessdescpl . . . . §9.2; 480, 680, 681
\glsaccessdescpl . . . . §9.2; 480, 680, 682
\GLSaccessdescpl . . . . §9.2; 480, 680, 682
\GLSaccessfirst . . . . §9.2; 478, 680, 682
\GLSaccessfirst . . . . §9.2; 478, 680, 682
\glsaccessfirst . . . . §9.2; 478, 680, 682
\GLSaccessfirstpl . . . . §9.2; 479, 680, 682
\GLSaccessfirstpl . . . . §9.2; 479, 680, 682
Index
\glsaccessfmttext . . . . . . . . §9.3; 485, 686
\GLSaccessfmtuseri . . . . . . . §9.3; 490, 686
\Glsaccessfmtuseri . . . . . . . §9.3; 489, 686
\glsaccessfmtuseri . . . . . . . §9.3; 489, 686
\GLSaccessfmtuserii . . . . . . §9.3; 490, 686
\Glsaccessfmtuserii . . . . . . §9.3; 490, 687
\glsaccessfmtuserii . . . . . . §9.3; 490, 687
\GLSaccessfmtuseriii . . . . . §9.3; 490, 687
\Glsaccessfmtuseriii . . . . . §9.3; 490, 687
\glsaccessfmtuseriii . . . . . §9.3; 490, 687
\GLSaccessfmtuseriv . . . . . . §9.3; 491, 687
\Glsaccessfmtuseriv . . . . . . §9.3; 491, 687
\glsaccessfmtuseriv . . . . . . §9.3; 490, 688
\GLSaccessfmtuserv . . . . . . . §9.3; 491, 688
\Glsaccessfmtuserv . . . . . . . §9.3; 491, 688
\glsaccessfmtuserv . . . . . . . §9.3; 491, 688
\GLSaccessfmtuservi . . . . . . §9.3; 491, 688
\Glsaccessfmtuservi . . . . . . §9.3; 491, 688
\glsaccessfmtuservi . . . . . . §9.3; 491, 688
\glsaccessibility . . . . . . . . . 688, 695, 761
\GLSaccesslong . . . . . . . §9.2; 481, 683, 689
\Glsaccesslong . . . . . . . §9.2; 481, 683, 689
\glsaccesslong . . . §9.2; 176, 481, 683, 689
\GLSaccesslongpl . . . . . §9.2; 482, 683, 689
\Glsaccesslongpl . . . . . §9.2; 482, 683, 689
\glsaccesslongpl . . . . . §9.2; 482, 683, 689
\GLSaccessname . . . . . . . §9.2; 477, 683, 690
\Glsaccessname . . . . . . . §9.2; 477, 683, 690
\glsaccessname . . . §9.2; 183, 477, 684, 690
\GLSaccessplural . . . . . §9.2; 478, 684, 690
\Glsaccessplural . . . . . §9.2; 478, 684, 690
\glsaccessplural . . . . . §9.2; 478, 684, 690
\GLSaccessshort . . . . . . §9.2; 481, 684, 690
\Glsaccessshort . . . . . . §9.2; 481, 684, 691
\glsaccessshort . . . . . §9.2; 177, 480, 684,
690, 691
\GLSaccessshortpl . . . . §9.2; 481, 685, 691
\Glsaccessshortpl . . . . §9.2; 481, 685, 691
\glsaccessshortpl . . . . §9.2; 481, 685, 691
\GLSaccesssymbol . . . . . §9.2; 479, 685, 691
\Glsaccesssymbol . . . . . §9.2; 479, 685, 691
\glsaccesssymbol . §9.2; 479, 685, 691, 692
\GLSaccesssymbolplural . . . . . . §9.2; 480,
685, 692

\Glsaccesssymbolplural . . . . . . §9.2; 479,
685, 692
\glsaccesssymbolplural . . . . . . §9.2; 479,
686, 692
\GLSaccesstext . . . . . . . §9.2; 478, 686, 692
\Glsaccesstext . . . . . . . §9.2; 478, 686, 692
\glsaccesstext . . . . §9.2; 15, 477, 686, 692
\GLSaccessuseri . . . . . . §9.2; 482, 686, 693
\Glsaccessuseri . . . . . . §9.2; 482, 686, 693
\glsaccessuseri . . . . . . §9.2; 482, 686, 693
\GLSaccessuserii . . . . . §9.2; 483, 687, 693
\Glsaccessuserii . . . . . §9.2; 482, 687, 693
\glsaccessuserii . . . . . §9.2; 482, 687, 693
\GLSaccessuseriii . . . . §9.2; 483, 687, 694
\Glsaccessuseriii . . . . §9.2; 483, 687, 694
\glsaccessuseriii . . . . §9.2; 483, 687, 694
\GLSaccessuseriv . . . . . §9.2; 483, 687, 694
\Glsaccessuseriv . . . . . §9.2; 483, 687, 694
\glsaccessuseriv . . . . . §9.2; 483, 688, 694
\GLSaccessuserv . . . . . . §9.2; 484, 688, 694
\Glsaccessuserv . . . . . . §9.2; 484, 688, 695
\glsaccessuserv . . §9.2; 483, 688, 694, 695
\GLSaccessuservi . . . . . §9.2; 484, 688, 695
\Glsaccessuservi . . . . . §9.2; 484, 688, 695
\glsaccessuservi . . . . . §9.2; 484, 688, 695
\glsaccsupp . . . . . . . . . . . . . . . . . . . . 695, 713
\glsacspace . . . . . . . . . . . . 132, 679, 695, 696
\glsacspacemax . . . . . . . . . . . . 132, 695, 696
\glsadd . . . 20, 23, 27, 37, 186–190, 196, 198,
259–261, 274, 502, 599, 616, 696, 697,
784, 934
\glsaddall 15, 186, 198, 259, 260, 376, 514,
618, 696
\glsaddallunindexed . §5.8; 259, 260, 696
\glsaddallunused . . . . . . 259, 260, 514, 696
\glsaddeach . . . . . . . . . . . §5.8; 198, 260, 696
\glsaddkey . . . . . . . . . . . . . . . 35, 696, see also
\glsaddstoragekey &
\glsxtrprovidestoragekey
\glsaddpostsetkeys . §5.1.1; 187, 188, 697
\glsaddpresetkeys . . §5.1.1; 187, 188, 697
\glsaddstoragekey . . 35, 697, 858, 859, see
also \glsaddkey &
\glsxtrprovidestoragekey

965


Index

\glsaltlistitem \ §8.6.5.3; 429, 697
\glsalttreechildpredesc \ §8.6.5.4; 431, 697
\glsalttreepredesc \ §8.6.5.4; 431, 697
\glsapptopostlink \ §5.5.4; 250, 698
\glsautoprefix \ 378, 698
\glsbackslash \ 378, 698
\glscapitalisewords \ §5.2.4; 200, 291, 373, 500, 698, 802
\glscapscase \ 254, 351, 352, 698, 817
\glscategory \ §11.5.2; 535, 698
\glscategorylabel \ §4.5.3.1; 162, 163, 166, 167, 699
\glscapturedgroup \ §11.5.2; 535, 698
\glscategory \ §10; 492, 699
\glscategorylabel \ §4.5.3.1; 162, 163, 166, 167, 699
\glscustomtext \ 254, 700, 732
\glsdefaultshortaccess \ 476, 700
\glsdefaulttype \ 44, 377, 627, 662, 667, 673, 701, 899, 925, 933
\glsdefpostdesc \ §8.6.2; 422, 701
\glsdefpostlink \ §5.5.4; 250, 701
\glsdefpostname \ §8.6.1; 420, 701
\GLSdesc \ 701
\Glsdesc \ 701
\glsdesc \ 252, 605, 701, 702
\glsdescplural \ 252, 702
\glsdescriptionaccessdisplay \ 480, 487, 702
\glsdescriptionpluralaccessdisplay \ 480, 488, 702
\glsdescwidth \ 446, 449, 450, 452, 454, 457, 458, 460, 702, 740, 743, 744, 747, 748
\Glsdisp \ 565, 702
\glsdisp \ 183, 235, 242, 254, 258, 366, 565, 600, 669, 700, 702, 703, 733, 799, 846, 913
\glsdisplaynumberlist \ §11.5.8; 571, 703
\glsdoifexists \ 703, 714, 894, 895, see also ifglsentryexists, glsdoifexistsordo & glsdoifnoexists
\glsdoifexistsordo \ 703, see also ifglsentryexists, glsdoifexists & glsdoifnoexists
\glsdoifexistsorwarn \ 703, see also ifglsentryexists, glsdoifexistsordo & glsdoifnoexists
\glsdoifnoexists \ 703, see also ifglsentryexists, glsdoifexistsordo & glsdoifnoexists
\glsenableentrycount \ 280, 307–309, 313, 501, 704, 708, 856, 857
\glsenableentryunitcount \ §6.1; 280, 313, 704, 708, 856, 857
\glsenacapwrcontent \ §5.8; 185, 264, 704
\glsentrycounterlabel \ 412, 480, 604, 605, 679, 705
\glsentrycurrcount \ 307, 313, 704, 708, 856, 857
\glsentrydesc \ 50, 208, 209, 229, 233, 481, 689, 706, 752
\glsentryfmt \ 159, 254, 494, 667, 706, 727, 780, 808, 859
\glsentryindexcount \ §5.8; 264, 706, 730
\glsentryitem \ 414, 704, 706
\Glsentrylong \ 50, 706
\glsentrylong \ 50, 208, 209, 229, 233, 481, 689, 706, 752
\Glsentrylongpl \ 50, 209, 210, 482, 689, 707, 753

966
Index

\Glsentryname .................. 374, 417, 707
\glsentryname .................. 23, 47, 212, 291, 292, 300, 374, 412, 414, 417, 477, 492, 574, 575, 604, 608, 690, 707
\glsentrynumberlist ........... \S 11.5.8; 436, 572, 707
\glsentryparent ............... 292, 707
\glsentrypdfsymbol .......... \S 8.6; 419, 708
\Glsentryplural ............... 708
\glsentryplural ............... 213, 478, 690, 708
\glsentryprevcount .......... 307, 314, 708
\glsentryprevmaxcount ....... \S 6.1; 314, 708
\glsentryprevtotalcount ....... \S 6.1; 314, 708
\Glsentryshort ................ 50, 709
\glsentryshort ............... 50, 201, 207, 216, 229, 230, 233, 481, 691, 709, 752
\Glsentryshortpl ............. 709
\glsentryshortpl ............. 50, 207, 208, 210, 481, 691, 709, 753
\Glsentrysymbol ............... 709
\glsentrysymbol ............... 419, 479, 610, 692, 709, 710
\glsentriesymbolaccess ...... 710
\glsentriesymbolplural ...... 479, 611, 692, 710
\Glsentrytext ................ 697, 710
\glsentrytext ............... 15, 50, 201, 212, 230, 300, 478, 693, 697, 710, 728
\glsentrytitlecase ........... 200, 374, 500, 710
\glsentrytype ................. 710
\Glsentryuseri ................. 711
\glsentryuseri ............... 482, 611, 693, 711
\Glsentryuserii .............. 711
\glsentryuserii ............. 482, 612, 693, 711
\Glsentryuseriiii ........... 711
\glsentryuseriiii ............ 483, 612, 694, 711
\Glsentryuseriv .............. 712
\glsentryuseriv ............. 483, 612, 694, 712
\Glsentryuserserv ........... 712
\glsentryuserserv ........... 484, 612, 695, 712
\Glsentryuserservi .......... 712
\glsentryuserservi .......... 484, 613, 695, 713
\glsexclapplyinnerfmtfield ... \S 5.5.3; 162, 245, 713
\glsexpandfields .............. 713, 760
\glsextrapostnamehook ....... \S 8.6.1; 421, 603, 713, 856
\glsfieldaccsupp .............. 713, 792
\glsfielddef ................. 37, 713, 714
\glsfieldeddef ............... 714
\glsfieldfetch ............... 710, 714
\glsfieldgdef ............... 714
\glsfieldxdef ............... 714
\glsFindWidestAnyName ...... \S 8.6.5.4; 435, 714, 715
\glsFindWidestAnyNameLocation \S 8.6.5.4; 436, 714
\glsFindWidestAnyNameSymbol \S 8.6.5.4; 436, 715
\glsFindWidestAnyNameSymbolLocation \S 8.6.5.4; 436, 715
\glsFindWidestLevelTwo ...... \S 8.6.5.4; 436, 715
\glsFindWidestTopLevelName ... \S 8.6.5.4; 435, 715
\glsfindwidesttoplevelname ... 435, 715
\glsFindWidestUsedAnyName ... \S 8.6.5.4; 435, 436, 715, 716
\glsFindWidestUsedAnyNameSymbol \S 8.6.5.4; 436, 716
\glsFindWidestUsedAnyNameSymbol-Location \S 8.6.5.4; 436, 716
\glsFindWidestUsedLevelTwo ... \S 8.6.5.4; 435, 716
\glsFindWidestUsedTopLevelName \S 8.6.5.4; 435, 436, 716
\GLSfirst ..................... 166, 213, 498, 717
\Glsfirst ..................... 166, 213, 498, 717
\glsfirst ................. 5, 43, 47, 50, 51, 61, 81, 166, 185, 186, 189, 213, 240, 282, 365, 498, 717, 864, 905
\glsfirstabbrvdefaultfont ... 65, 74, 81, 93, 115, 132, 717
\glsfirstabbrvdefaultfont ... 71–73, 78–80, 85–87, 97–99, 126–129, 158, 717
\glsfirstabbrvdefaultfont ... 166, 167, 171, 672, 717, 719, 721

967
Index

Index
Index

\glsentryfmt \ §5.5.5; 60, 65, 81, 191, 243, 255, 484, 494, 706, 727
\glsgetattribute \ §10.2.2; 505, 727
\glsgetcategoryattribute \ §10.2.2; 505, 727
\glsgetwidestname \ §8.6.5.4; 434, 727
\glsgetwidestsubname \ §8.6.5.4; 435, 727
\glsgroupheading \ 393, 398–401, 727
\glsgroupskip \ 383, 428, 728
\glshascategoryattribute \ §10.2.2; 354, 505, 728
\glsclasscategoryattribute \ §10.2.2; 354, 505, 728
\glsclasshashchar \ §11.5.2; 535, 728
\glsclasshex \ §11.5.2; 535, 728
\glsclasshyperlink \ 728
\glsclasshypernumber \ 29, 556, 560, 576, 729, 752, 894, 933
\glsifapplyinnerfmtfield \ §5.5.3; 244, 729, see also
\glsifexclapplyinnerfmtfield
\glsifattribute \ §10.2.2; 505, 729
\glsifattributetrue \ §10.2.2; 506, 729
\glsifcategory \ §10; 492, 729
\glsifcategoryattribute \ §10.2.2; 505, 506, 729
\glsifcategoryattributehasitem \ §10.2.2; 507, 730
\glsifcategoryattributetrue \ §10.2.2; 506, 730
\glsifindexed \ §5.8; 264, 730
\glsifndef \ §10.2.2; 506, 730
\glsifndefregular \ §10.2.2; 506, 730
\glsifndefregularcategory \ §10.2.2; 506, 730
\glsclassplural \ 142, 144, 150, 152, 254, 351, 352, 730
\glsclassregular \ §10.2.2; 255, 506, 706, 731
\glsclassregularcategory \ §10.2.2; 506, 731
\glsclassignore \ 261, 424, 525, 601, 731
\glsclassindexingsetting \ 375, 510, 731
\glsclassindex subgroupitem \ 731
\glsclassintreunsets \ §5.1.1; 186, 189, 286, 731
\glsclasslinedescformat \ 428, 732
\glsclasslinesubdescformat \ 428, 732
\glsclassinnerfmtabbrvfont \ §4.5.3.1; 167, 168, 732, 778
\glsclassinnerfmtlongfont \ §4.5.3.1; 168, 721, 732, 779
\glsclassinsert \ 54, 105, 150–152, 166, 253, 254, 723, 732, 807, 861, see also
\glsclassxtrasaveinsert & \glsclassxtrfullsaveinsert
\glsclasskeylisttok \ §4.5.3.1; 161–163, 732
\glsclasslabel \ 150–152, 165, 166, 230, 243, 246, 254, 260, 286, 502, 732, 784, 855
\glsclasslabeltok \ §4.5.3.1; 161, 163–166, 733
\glsclassletentryfield \ 291, 733
\glsclasslink \ 565, 733
\glsclasslink \ 183, 235, 242, 252, 258, 296, 565, 600, 670, 703, 733, 803, 847
\glsclasslinkcheckfirsthyperhook \ 186, 307, 316, 319, 733
\glsclasslinkpostsetkeys \ 187–190, 307, 320, 733
\glsclasslinkpresetkeys \ §5.1.1; 187–190, 307, 319, 320, 533, 734
\glsclasslinkwrccontent \ §5; 185, 230, 704, 734
\glsclasslistchildpostlocation \ §8.6.5.3; 429, 734
\glsclasslistchildprelocation \ §8.6.5.3; 428, 734
\glsclasslistdesc \ §8.6.5.3; 429, 734
\glsclasslistdottedwidth \ 429, 734
\glsclasslistexpandedname \ 734
\glsclasslistgroupafterheader \ §8.6.5.3; 429, 735
\glsclasslistgroupheaderitem \ §8.6.5.3; 429, 735
\glsclasslistgroupskip \ §8.6.5.3; 430, 735
\glsclasslistinit \ 735
\glsclasslistitem \ §8.6.5.3; 429, 735
\glsclasslistprelocation \ §8.6.5.3; 428, 429, 735
\glsclasslocalreset \ 191, 280, 735, 856, see also
\glsreset, \glslocalunset,
\glsifglsused & \GlsXtrIfUnusedOrUndefined
\glsclasslocalresetall \ 287, 736, see also

969
Index

TabularFooter .... §8.7.2.3; 453, 741
\glslongextraShortNoNameSetDescWidth .......... §8.7.2.6; 460, 744
\glslongextraNameDescLocationTabularFooter .... §8.7.2.3; 453, 741
\glslongextraNameDescSymHeader §8.7.2.2; 451, 741
\glslongextraNameDescSymLocation-TabularFooter §8.7.2.4; 454, 742
\glslongextraNameDescSymLocation-TabularHeader §8.7.2.4; 454, 742
\glslongextraNameDescSymTabularFooter §8.7.2.2; 450, 742
\glslongextraNameDescSymTabularHeader §8.7.2.2; 450, 742
\glslongextraNameFmt .... §8.7.2; 446, 742
\glslongextraNameSymDescHeader §8.7.2.2; 451, 743
\glslongextraNameSymDescLocation-TabularFooter §8.7.2.4; 455, 743
\glslongextraNameSymDescLocation-TabularHeader §8.7.2.4; 455, 743
\glslongextraNameSymDescLocation-TabularFooter §8.7.2.4; 454, 743
\glslongextraNameSymDescTabularFooter §8.7.2.2; 451, 743
\glslongextraSetDescWidth .... §8.7.2.1; 449, 450, 452, 743
\glslongextraSetWidest §8.7.2; 447, 449, 450, 452, 454, 744
\glslongextraShortHeader §8.7.2.6; 458, 460, 744
\glslongextraShortLongHeader §8.7.2.6; 460, 744
\glslongextraShortLongTabularFooter §8.7.2.6; 460, 744
\glslongextraShortLongTabularHeader §8.7.2.6; 460, 744
\glslongextraShortNoNameSetDescWidth .......... §8.7.2.6; 460, 744
\glslongextraShortTargetFmt §8.7.2.6; 459, 744
\glslongextraSubDescFmt ...... §8.7.2; 446, 744
\glslongextraSubGroupHeading §8.7.2; 448, 745
\glslongextraSubLocationFmt .... §8.7.2; 447, 745
\glslongextraSubLongFmt .... §8.7.2.6; 460, 745
\glslongextraSubNameFmt .... §8.7.2; 446, 447, 456, 745
\glslongextraSubShortTargetFmt §8.7.2.6; 459, 745
\glslongextraSubSymbolFmt ...... §8.7.2; 448, 745
\glslongextraSubSymbolOrName §8.7.2.5; 457, 746
\glslongextraSubSymbolTargetFmt §8.7.2.5; 456, 457, 746
\glslongextraSymbolAlign ...... §8.7.2; 448, 746
\glslongextraSymbolFmt ...... §8.7.2; 448, 456, 746
\glslongextraSymbolNameAlign §8.7.2.5; 456, 746
\glslongextraSymbolOrName .... §8.7.2; 456, 746
\glslongextraSymbolTargetFmt §8.7.2.5; 456, 746
\glslongextraSymDescHeader .... §8.7.2.5; 457, 747
\glslongextraSymDescNameHeader §8.7.2.2; 452, 747
\glslongextraSymDescNameTabularFooter .... §8.7.2.2; 451, 747
\glslongextraSymDescNameTabularHeader .... §8.7.2.2; 451, 747
\glslongextraSymDescTabularFooter §8.7.2.5; 457, 747
\glslongextraSymDescTabularHeader §8.7.2.5; 457, 747
Index

\glslongextraSymLocSetDescWidth §8.7.2.4; 454, 747
\glslongextraSymNoNameSetDescWidth §8.7.2.5; 457, 747
\glslongextraSymSetDescWidth §8.7.2.2; 450, 454, 748
\glslongextraTabularVAlign §8.7.2; 447, 748
\glslongextraUpdateWidest §8.7.2; 447, 748
\glslongextraUpdateWidestChild §8.7.2; 447, 748
\GlsLongExtraUseTabularfalse §8.7.2; 445, 748
\GlsLongExtraUseTabulartrue §8.7.2; 445, 748
\glslongfont 168, 172, 732, 749, 779
\glslongfootnotefont 115, 145, 749
\glslonghyphenfont 64, 103, 108, 148, 749
\glslongonlyfont 111, 154, 749
\glslongpltok §4.5.3.1; 162, 164, 749
\glslongpluralaccessdisplay 482, 489, 749
\glslongtok §4.5.3.1; 156, 161, 164, 749
\glslonguserfont 88, 100, 136, 749
\glslowercase §5.2.2; 200, 750
\glsmakefirstuc 750
\glsmfuadmap §5.2.1; 199, 292, 567, 750
\glsmfublocker §5.2.1; 199, 567, 750
\glsmfuxcl §5.2.1; 198, 199, 750
\GLSname 81, 89, 91, 93, 101, 103, 212, 750
\Glsname 212, 365, 751, 909
\glnames 212, 352, 365, 608, 750, 751, 909
\glnameaccessdisplay 477, 485, 751
\glnamefont 415, 419, 458, 473, 501, 751
\glnextpages 415, 751
\glnoexpandfields 205, 553, 751, 760
\glnoiddisplayloc §11.5.6; 424, 556, 751, 797
\glnonextpages 415, 752, 757
\glnumberformat 188, 602, 752
\glnumbersgroupname 426, 752, 948
\glspagelistwidth 447, 452, 454, 752
\glspar 752
\glspdfmftrue §5.3.2; 210, 211, 722, 723, 752
\glspdfmtfull §5.3.2; 210, 211, 723, 753
\glspercentslash 502, 753
\GLSp1 753
\Glspl 753
\glsp1 753
\Glsplural 213, 753
\Glsplural 43, 47, 81, 185, 191, 213, 240, 252, 495, 731, 753, 754
\glspluralaccessdisplay 478, 485, 754
\glsplursuffix 35, 36, 43, 44, 162, 606, 608, 754
\glspostdescription 12, 13, 421–423, 427, 428, 603, 605, 754, 853
\glspostinline 428, 754
\glsplinedescformat 755
\glsplinesubdescformat 755
\glsplinkhook 246, 347, 349, 603, 755
\glsprefixsep 55, 755
\glsprestandardsort 755, 949
\glsprepostlink §5.5.4; 250, 755
\glsp §5.5.4; 231, 233, 332, 756
\glsp §5.4; 231, 756
\glsrefentry 377, 416, 756, 851
\glsrenewcommand §11.5.8; 572, 756
\glsreset 191, 280, 307, 756, 857, see also
\glslocalreset, \glsunset, \ifglsused & \GlsXtrIfUnusedOrUndefined
\glsresetall 280, 756, see also \glsreset,
\glsunsetall, \ifglsused & \GlsXtrIfUnusedOrUndefined
\glsresetcurrcountfalse §6.1; 307, 308, 756
\glsresetcurrcounttrue §6.1; 308, 757
\glsresetentrylist 757
\glssee 21, 22, 24, 33, 37, 186, 259, 265–267, 270, 272, 278, 609, 757, see also
\glsxtrindexseealso
\glsseefirstitem §5.13; 300, 757, 758, see also \glsseeitem
Index

\glsseeformat ................. 278, 299, 757, 892
\glsseeitem .... 299–301, 757, 758, see also
\glsseefirstitem
\glsseeitemformat ........ \S 5.13; 292, 300, 757
\glsseeelastxofoxdsep \S 5.13; 301, 758, see also \glsseeelastsep & \glsseeelastxofoxdsep
\glsseeelastsep \S 5.13; 301, 662, 758, see also \glsseeelastsep & \glsseeelastxofoxdsep
\glsseeelist \S 5.13; 299, 300, 302, 757, 758, 863, 877, see also \glsxtrseeelist
\glsseeseep \S 5.13; 301, 758, see also \glsseeseep & \glsseeelastxofoxdsep
\glssetattribute ........ \S 10.2.2; 504, 759
\glssetcategoriesattribute \S 10.2.2; 504, 759
\glssetcategoriesattributes \S 10.2.2; 504, 759
\glssetcategoryattribute \S 10.2.2; 504, 754, 759
\glssetcategoryattributes \S 10.2.2; 504, 759
\glssetcombinedsepabbrvnbsp \S 7.4; 345, 759
\glssetcombinedsepabbrvnone \S 7.4; 345, 759
\glssetcombinedsepnarrow \S 7.4; 346, 760
\glssetexpandfield .......... 751, 760
\glssetnoexpandfield .......... 713, 760
\glssetregularcategory \S 10.2.2; 504, 760
\glssetwidest 434, 447, 461, 554, 645, 671, 674, 760, 774
\glsshortaccessdisplay .... 481, 488, 760
\glsshortaccsupp .......... 713, 760, 858
\glsshortpltok \S 4.5.3.1; 162, 164, 167, 761
\glsshortpluralaccessdisplay .... 481, 488, 761
\glsshorttok \S 4.5.3.1; 145, 161, 162, 164, 167, 761
\glsshowtarget .................. 761
\glsshowtargetfont ................. 761
\glsshowtargetfonttext ........ 28, 761
\glsshowtargetinner 29, 30, 761, 762, 872
\glsshowtargetinnersymleft \S 2.5; 30, 762
\glsshowtargetinnersymright \S 2.5; 30, 762
\glsshowtargetouter 29, 30, 761, 762, 873
\glsshowtargetsymbol ............. 762
\glssubentryitem ................. 414, 762
\glssubplotheading \S 8.4.1; 393, 394, 400, 401, 762
\GLSsymbol ..................... 763
\Glsymbol ..................... 763
\glssymbol .................... 365, 366, 610, 763, 912
\glssymbolaccessdisplay ......... 416, 479, 486, 763
\glssymbolplural ............... 611, 763
\glssymbolpluralaccessdisplay ... 479, 487, 763
\glsymplgroupname .............. 426, 764, 950
\glstabledesheader \S 8.7.4; 469, 764
\glstableiffilter \S 8.7.4; 468, 764
\glstablenameheader \S 8.7.4; 469, 764
\glstableotherheader \S 8.7.4; 470, 764
\glstablesymbolheader \S 8.7.4; 469, 764
\glsstartag 29, 256, 379, 415, 502, 764
\GLText .................. 204, 212, 697, 698, 765
\Gltext .................. 212, 697, 698, 765
\gltext 5, 23, 37, 43, 47, 51, 81, 183, 185, 191, 192, 201, 204, 212, 234, 240, 244, 252, 282, 319, 364, 495, 511, 600, 697, 698, 765, 908, 911
\gltext-like .................. 600
\gltextaccessdisplay .... 478, 485, 765
\gltextformat .... 190, 191, 238, 239, 254, 499, 616, 618, 763, 784
\gltextup ................. 157, 765, 862
\glstildechar .................. 502, 766
\glstildecharassignsubindent \S 8.7.3; 467, 766
Index

\glstopicAssignWidest \texttt{§8.7.3}; 467, 766 \glstreeNoDescSymbolPreLocation
\glstopicCols \texttt{§8.7.3}; 462, 766 \texttt{§8.6.5.4}; 433, 771
\glstopicColsEnv \texttt{§8.7.3}; 462, 766 \glstreenavigationfmt \texttt{§8.6.5.4}; 430, 771
\glstopicDesc \texttt{§8.7.3}; 466, 766 \glstreenavigationfmt \texttt{§8.6.5.4}; 432, 771
\glstopicGroupHeading \texttt{§8.7.3}; 465, 766 \glstreenamedesc \texttt{§8.6.5.4}; 432, 771
\glstopicInit \texttt{§8.7.3}; 465, 766 \glstreeitem \texttt{§8.6.5.4}; 432, 771
\glstopicItem \texttt{§8.7.3}; 465, 466, 767 \glstreeitem \texttt{§8.6.5.4}; 432, 771
\glstopicLoc \texttt{§8.7.3}; 467, 767 \glstreenamedescloc \texttt{§8.6.5.4}; 433, 771
\glstopicMarker \texttt{§8.7.3}; 466, 767 \glstreepredesc \texttt{§8.6.5.4}; 431, 772
\glstopicMidSkip \texttt{§8.7.3}; 466, 767 \glstreePreHeader \texttt{§8.6.5.4}; 431, 772
\glstopicParIndent \texttt{§8.7.3}; 463, 767 \glstreeprelocation \texttt{§8.6.5.4}; 432, 772
\glstopicPostSkip \texttt{§8.7.3}; 466, 767 \glstreesubgroupitem \texttt{§8.7.3}; 772
\glstopicSubGroupHeading \texttt{§8.7.3}; 465, 768 \glstreesubitem \texttt{§8.7.3}; 442, 772
\glstopicSubIndent \texttt{§8.7.3}; 464, 768 \glstreesubheader \texttt{§8.7.3}; 772
\glstopicSubItem \texttt{§8.7.3}; 467, 768 \glstreesubitem \texttt{§8.7.3}; 442, 772
\glstopicSubItemBox \texttt{§8.7.3}; 467, 768 \glstreesymbol \texttt{§8.6.5.4}; 432, 773
\glstopicSubItemParIndent \texttt{§8.7.3}; 464, 768 \glstriggerrecordformat \texttt{§11.4}; 525, 773
\glstopicSubItemSep \texttt{§8.7.3}; 467, 768 \texttt{§5.2.3}; 200, 774, 900
\glstopicSubLoc \texttt{§8.7.3}; 468, 768 \texttt{§4.5.2}; 159, 774
\glstopicSubNameFont \texttt{§8.7.3}; 467, 768 \texttt{§4.5.2}; 159, 774
\glstopicSubPreLocSep \texttt{§8.7.3}; 468, 769 \texttt{§4.5.2}; 159, 774
\glstopicTitle \texttt{§8.7.3}; 466, 769 \texttt{§4.5.2}; 159, 774
\glstopicTitleFont \texttt{§8.7.3}; 466, 769 \texttt{§4.5.2}; 159, 774
\glstreechilddesc \texttt{§8.6.5.4}; 433, 769 \texttt{§4.5.2}; 159, 774
\glstreeChildDescLoc \texttt{§8.6.5.4}; 433, 769 \glsunset \texttt{§8.6.5.4}; 434, 772
\glstreechildpredesc \texttt{§8.6.5.4}; 431, 769 \glsunsetall \texttt{§10.2.2}; 504, 773
\glstreechildprelocation \texttt{§8.6.5.4}; 432, 433, 769 \glsupdate widest \texttt{§8.6.5.4}; 434, 672, 774
\glstreechildsymbol \texttt{§8.6.5.4}; 433, 770 \glsunsetcategoryattribute \texttt{§10.2.2}; 504, 773
\glstreedefaultnamefmt \texttt{§8.6.5.4}; 430, 770 \glsupercase \texttt{§5.2.3}; 200, 774, 900
\glstreedesc \texttt{§8.6.5.4}; 432, 433, 770 \glsuseabbrvfont \texttt{§4.5.2}; 159, 774
\glstreeDescLoc \texttt{§8.6.5.4}; 432, 433, 770 \glsuselongfont \texttt{§4.5.2}; 159, 774
\glstreegroupheaderfmt \texttt{§8.6.5.4}; 430, 443, 770 \texttt{§8.6.5.4}; 434, 672, 774
\glstreegroupheaderskip \texttt{§8.6.5.4}; 430, 431, 770 \texttt{§8.6.5.4}; 434, 672, 774
\glstreegroupskip \texttt{§8.6.5.4}; 430, 431, 770 \texttt{§8.6.5.4}; 434, 672, 774
\glstreesubitem \texttt{§8.6.5.4}; 770 \texttt{§8.6.5.4}; 434, 672, 774
\glstreenamefmt \texttt{§8.6.5.4}; 430, 771 \texttt{§8.6.5.4}; 434, 672, 774
\glstreenavigationfmt \texttt{§8.6.5.4}; 430, 771 \texttt{GLSuseri}; 774
\glsuseri \texttt{§8.6.5.4}; 434, 672, 774
\glsuserii \texttt{§8.6.5.4}; 434, 672, 774
\glsuseriaccessdisplay \texttt{§8.6.5.4}; 434, 672, 774
\glsuseriaccessdisplay \texttt{§8.6.5.4}; 434, 672, 774
\glsuseriiaccessdisplay \texttt{§8.6.5.4}; 434, 672, 774
\glsupdatewidest \texttt{§8.6.5.4}; 434, 672, 774

974
Index

\glsuseriiaccessdisplay . 482, 490, 775 \glsxtrAccSuppAbbrSetNoLongAttrs §4.5.3.1; 168, 781
\GLSuserii . 776
\Glsuserii . 776
\glsuserii . 612, 776
\glsuseriiaccessdisplay . 483, 490, 776
\GLSuseriv . 776
\Glsuseriv . 776
\glsuseriv . 612, 776, 777
\glsuserivaccessdisplay . 483, 491, 777
\GLsuserv . 777
\Glsuserv . 777
\glsuserv . 612, 777
\glsuservaccessdisplay . 484, 491, 777
\GLsuservi . 778
\Glsuservi . 778
\glsuservi . 613, 778
\glsuserviaccessdisplay . 484, 491, 778
\glswriteentry §5.8; 264, 778
\glsxpaabbrvfont §4.5.3.1; 168, 778
\glsxplongfont §4.5.3.1; 168, 779

Glsxtr
\Glsxtr §13; 579, 580, 779
\glsxtr §13; 578–580, 779, 791, 799, 852
\glsxtr@counterrecord §8.4.3.2; 407, 408, 779, 782
\glsxtr@record §11.5.6; 555, 779
\glsxtr@record@nameref §11.5.6; 555, 556, 779
\glsxtr@resource §11; 17, 511, 780
\glsxtr@abbreviationfont §5.5.2; 61, 65, 236, 240, 255, 706, 780
\glsxtrabbrvfootnote . 145, 780
\glsxtrabbrvpluralsuffix §4.1.2; 44, 136, 148, 154, 158, 171, 476, 654, 658, 780
\glsxtrabbrvtype §4.1.4; 9–11, 41, 44, 658, 780, 926, 927, 941
\glsxtrAccSuppAbbrSetFirstLongAttrs §4.5.3.1; 169, 780
\glsxtrAccSuppAbbrSetNameLongAttrs §4.5.3.1; 169, 780
\glsxtrAccSuppAbbrSetNameShortAttrs §4.5.3.1; 169, 781
\glsxtractivatenopost §8.5; 415, 417, 423, 781
\glsxtractualanchor §11.5.6; 557, 781, 863
\glsxtraddallcrossrefs §5.9.3; 21, 279, 781, 884, see also indexcrossrefs
\glsxtrAddCounterRecordHook §8.4.3.2; 407, 411, 779, 782
\glsxtraddgroup §8.4.1; 391, 392, 396, 401, 782
\glsxtraddlabelprefix §11.5.7; 563, 571, 782
\glsxtraddpunctuationmark §5.5.4; 249, 782, 822
\glsxtraddunusedxrefs §5.9.3; 279, 782
\glsxtralias §5.9.2; 36, 37, 277, 782
\glsxtraliashook §3.4; 36, 783
\glsxtrAltTreeIndent §8.6.5.4; 437, 783
\glsxtralttreeInit §8.6.5.4; 437, 783
\glsxtralttreeSubSymbolDescLocation §8.6.5.4; 437, 783
\glsxtralttreeSymbolDescLocation §8.6.5.4; 437, 783
\glsxtrapptocsvfield §3.5; 38, 299, 303, 783
\GlsXtrAppToDefaultGlsOpts §5.1.1; 186, 187, 783
\glsxtrassignactualsetup §9.1; 475, 476, 784
\glsxtrassignfieldfont §5.5.2; 159, 244, 784, 859
\glsxtrassignlinktextfmt §5.5.4; 160, 254, 784
\glsxtrattrentrytextfmt §5.5.3; 243, 784
\GlsXtrAutoAddOnFormat §5.8; 260, 261, 784
\glsxtrautoindex §12; 575, 784, 797
\glsxtrautoindexassignsort §12; 575, 784
## Index

<table>
<thead>
<tr>
<th>Command</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>\glsxtrautoindexentry</td>
<td>§12; 574, 785</td>
</tr>
<tr>
<td>\glsxtrautoindexesc</td>
<td>§12; 575, 785</td>
</tr>
<tr>
<td>\glsxtrBasicDigitrules</td>
<td>545, 785</td>
</tr>
<tr>
<td>\glsxtrbibaddress</td>
<td>§11.5.2; 536, 785</td>
</tr>
<tr>
<td>\glsxtrbibauthor</td>
<td>§11.5.2; 536, 785</td>
</tr>
<tr>
<td>\glsxtrbibbooktitle</td>
<td>§11.5.2; 536, 785</td>
</tr>
<tr>
<td>\glsxtrbibchapter</td>
<td>§11.5.2; 536, 785</td>
</tr>
<tr>
<td>\glsxtrbibeducation</td>
<td>§11.5.2; 536, 785</td>
</tr>
<tr>
<td>\glsxtrbibhowpublished</td>
<td>§11.5.2; 536, 786</td>
</tr>
<tr>
<td>\glsxtrbibinstitution</td>
<td>§11.5.2; 536, 786</td>
</tr>
<tr>
<td>\glsxtrbibjournal</td>
<td>§11.5.2; 536, 786</td>
</tr>
<tr>
<td>\glsxtrbibmonth</td>
<td>§11.5.2; 536, 786</td>
</tr>
<tr>
<td>\glsxtrbibnote</td>
<td>§11.5.2; 536, 786</td>
</tr>
<tr>
<td>\glsxtrbibnumber</td>
<td>§11.5.2; 536, 786</td>
</tr>
<tr>
<td>\GlsXtrBibTeXEntryAliases</td>
<td>§11.5.2; 535, 787</td>
</tr>
<tr>
<td>\glsxtrbiborganization</td>
<td>§11.5.2; 536, 786</td>
</tr>
<tr>
<td>\glsxtrbibpages</td>
<td>§11.5.2; 536, 787</td>
</tr>
<tr>
<td>\glsxtrbibpublisher</td>
<td>§11.5.2; 536, 787</td>
</tr>
<tr>
<td>\glsxtrbibschool</td>
<td>§11.5.2; 536, 787</td>
</tr>
<tr>
<td>\glsxtrbibseries</td>
<td>§11.5.2; 536, 787</td>
</tr>
<tr>
<td>\GlsXtrBibTeXEntryAliases</td>
<td>§11.5.2; 535, 787</td>
</tr>
<tr>
<td>\glsxtrbibtitle</td>
<td>§11.5.2; 536, 787</td>
</tr>
<tr>
<td>\glsxtrbibtype</td>
<td>§11.5.2; 536, 787</td>
</tr>
<tr>
<td>\glsxtrbibvolume</td>
<td>§11.5.2; 536, 788</td>
</tr>
<tr>
<td>\glsxtrbookindexatendgroup</td>
<td>§8.7.1; 441, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexbookmark</td>
<td>§8.7.1; 441, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexbetween</td>
<td>§8.7.1; 440, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexbookmark</td>
<td>§8.7.1; 442, 788</td>
</tr>
<tr>
<td>\glsxtrbookindexcols</td>
<td>§8.7.1; 438, 788</td>
</tr>
<tr>
<td>\glsxtrbookindexcolspread</td>
<td>§8.7.1; 439, 788</td>
</tr>
<tr>
<td>\glsxtrbookindexfirstmark</td>
<td>§8.7.1; 443, 788</td>
</tr>
<tr>
<td>\glsxtrbookindexfirstmarkfmt</td>
<td>§8.7.1; 444, 788, 789</td>
</tr>
<tr>
<td>\glsxtrbookindexformatheader</td>
<td>§8.7.1; 443, 789</td>
</tr>
<tr>
<td>\glsxtrbookindexformatsubheader</td>
<td>§8.7.1; 394, 789</td>
</tr>
<tr>
<td>\glsxtrbookindexlastmark</td>
<td>§8.7.1; 443, 789</td>
</tr>
<tr>
<td>\glsxtrbookindexlastmarkfmt</td>
<td>§8.7.1; 444, 789</td>
</tr>
<tr>
<td>\glsxtrbookindexlocation</td>
<td>§8.7.1; 440, 789</td>
</tr>
<tr>
<td>\glsxtrbookindexmarkentry</td>
<td>§8.7.1; 443, 789</td>
</tr>
<tr>
<td>\glsxtrbookindexmulticolsenv</td>
<td>§8.7.1; 439, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexname</td>
<td>§8.7.1; 439, 440, 444, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexparentchildsep</td>
<td>§8.7.1; 440, 441, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexparentschildsep</td>
<td>§8.7.1; 440, 441, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexpregroupskip</td>
<td>§8.7.1; 443, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexprelocation</td>
<td>§8.7.1; 440, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexsubatendgroup</td>
<td>§8.7.1; 441, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexsubbetween</td>
<td>§8.7.1; 441, 790</td>
</tr>
<tr>
<td>\glsxtrbookindexsublocation</td>
<td>§8.7.1; 440, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexsubname</td>
<td>§8.7.1; 440, 444, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexsubprelocation</td>
<td>§8.7.1; 440, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexsubsubatendgroup</td>
<td>§8.7.1; 441, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexsubsubbetween</td>
<td>§8.7.1; 441, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexsubsublocation</td>
<td>§8.7.1; 441, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexsubsubname</td>
<td>§8.7.1; 441, 791</td>
</tr>
<tr>
<td>\glsxtrbookindexsubsubprelocation</td>
<td>§8.7.1; 441, 791</td>
</tr>
<tr>
<td>\glsxtrchecknohyperfirst</td>
<td>§5.1.1; 186, 189, 498, 792</td>
</tr>
<tr>
<td>\glsxtrclearlabelprefixes</td>
<td>§11.5.7; 439, 791</td>
</tr>
<tr>
<td>\glsxtrchecknohyperfirst</td>
<td>§5.1.1; 186, 189, 498, 792</td>
</tr>
<tr>
<td>\glsxtrclearlabelprefixes</td>
<td>§11.5.7; 563, 792, see also \glsxtrdaddlabelprefix &amp; \glsxtrprependlabelprefix</td>
</tr>
<tr>
<td>\GlsXtrClearUnsetBuffer</td>
<td>§5.10.1; 285, 286, 792</td>
</tr>
</tbody>
</table>
Index

\glsxtrGeneralLatinIIIrules . 547, 808
\glsxtrGeneralLatinIIrules .547, 809
\glsxtrGeneralLatinIrules
546–548, 809
\glsxtrGeneralLatinIVrules . . .547, 809
\glsxtrGeneralLatinNtoSrules 548, 809
\glsxtrGeneralLatinNtoZrules 548, 809
\glsxtrGeneralLatinTtoZrules 548, 809
\glsxtrGeneralLatinVIIIrules . .547, 809
\glsxtrGeneralLatinVIIrules . 547, 809
\glsxtrGeneralLatinVIrules . . .547, 810
\glsxtrGeneralLatinVrules . . . .547, 810
\glsxtrgeneralpuncaccentsrules 544, 810
\glsxtrgeneralpuncbracketrules 544, 810
\glsxtrgeneralpuncIIrules . . . .545, 810
\glsxtrgeneralpuncIrules . 544, 810
\glsxtrgeneralpuncmarksrules 544, 810
\glsxtrgeneralpuncquoterules 544, 810
\glsxtrgeneralpuncrules . . . . . .543, 811
\glsxtrgeneralpuncsignrules . 545, 811
\glsxtrgetgrouptitle . . . . . . . §8.6.4; 426, 465, 811
\glsxtrgroupfield . §8.4.1; 389–391, 811
\glsxtrglossentry . §8.5; 414, 417, 422, 811, 874
\glsxtrglossentryother . §8.5; 415, 417, 811, 874
\GLSxtrheadfirst §5.3.3; 226, 811, see also
\GLSfmtfirst & \GLSxtrtitlefirst
\GLSxtrheadfirst §5.3.3; 225, 812, see also
\GLSfmtfirst & \GLSxtrtitlefirst
\GLSxtrheadfirst §5.3.3; 225, 812, see also \GLSfmtfirstpl & \GLSxtrtitlefirstpl
\GLSxtrheadfirstplural §5.3.3; 226, 812, see also \GLSfmtfirstpl & \GLSxtrtitlefirstplural
\GLSxtrheadfull §5.3.3; 222, 812, see also
\GLSfmtfull & \GLSxtrtitlefull
\GLSxtrheadfullpl §5.3.3; 223, 813, see also \GLSfmtfullpl & \GLSxtrtitlefullpl
\GLSxtrheadlong . . . . . . . . . . .  §5.3.3; 220, 813
\GLSxtrheadlong . . . . . . . . . . .  §5.3.3; 220, 813
\GLSxtrheadlong . . . . . . . . . . .  §5.3.3; 220, 813
\GLSxtrheadlongpl . . . . . . . . . . . . §5.3.3; 221, 814
\GLSxtrheadlongpl . . . . . . . . . . . . §5.3.3; 221, 814
\GLSxtrheadlongpl . . . . . . . . . . . . §5.3.3; 221, 814
\GLSxtrheadname . §5.3.3; 223, 814, see also
\GLSfmtname & \GLSxtrtitlename
\GLSxtrheadname . §5.3.3; 223, 814, see also \GLSfmtname & \GLSxtrtitlename
\GLSxtrheadfirst §5.3.3; 225, 812, see also \GLSfmtfirst & \GLSxtrtitlefirst
\GLSxtrheadplural . . . . . . . . . .  §5.3.3; 225, 815, see also \GLSfmtplural & \GLSxtrtitleplural
\GLSxtrheadplural . . . . . . . . . .  §5.3.3; 225, 815, see also \GLSfmtplural & \GLSxtrtitleplural
\GLSxtrheadplural §5.3.3; 226, 812, see also \GLSfmtfirstpl & \GLSxtrtitlefirstplural

979
Index

\glsxtritletential ....... §5.3.3; 218, 815
\GLSxtrheadshort ....... §5.3.3; 217, 218, 815
\Glsxtrheadshort ....... §5.3.3; 217, 815
\GLSxtrheadshortpl ....... §5.3.3; 220, 815
\Glsxtrheadshortpl ....... §5.3.3; 219, 815
\glsxtrheadshortpl ....... §5.3.3; 219, 816
\GLSxtrheadtext ....... §5.3.3; 224, 816, see also §5.3.3; 224, 816, see also
\GLSfmttext & \Glsxtrtitletext
\Glsxtrheadtext ....... §5.3.3; 224, 816, see also §5.3.3; 224, 816, see also
\GLSfmttext & \Glsxtrtitletext
\glsxtrheadtext ....... §5.3.3; 224, 816, see also §5.3.3; 224, 816, see also
\GLSfmttext & \Glsxtrtitletext
\GLXTRhiername ....... §5.11; 293, 816
\GLSxtrhiername ....... §5.11; 293, 816
\Glsxtrhiername ....... §5.11; 292, 816
\GLSxtrhiername ....... §5.11; 292, 817
\Glsxtrhiernamesselp ....... §5.11; 292, 817
\glsxtrhyphensrules ....... 543, 817
\glsxtrhyphensuffix ....... 148, 817
\glsxstripyfylslike ....... §5.7; 258, 817
\glsxtrifallcaps ....... §5.5.4; 253, 817
\glsxtrifcounttrigger ....... §6.1; 311, 501, 817
\glsxtrifcustomdiscardperiod ....... §5.5.4; 248, 818
\glsxtrifemptyglossary ....... 818, see also \ifglossaryexists & \glsxtrifemptyglossary
\GlsxtrIfFieldCmpNum ....... §5.15; 305, 306, 818, 819
\GLSxtrIfFieldEqNum ....... §5.15; 306, 818
\GlsxtrIfFieldEqStr ....... §5.15; 306, 818, 819, 824
\GlsxtrIfFieldEqXpStr ....... §5.15; 306, 819
\GlsxtrIfFieldNonZero ....... §5.15; 306, 554, 819, 820
\GlsxtrIfFieldUndef ....... §5.15; 305, 819, see also \ifglsfieldvoid
\GlsxtrIfValueInCsvList ....... §5.13; §5.15; 277, 278, 292, 301–306, 554, 801, 805, 818, 819, 822
\GlsxtrIfHasNonZeroChildCount ....... §11.5.4; 554, 820
\GlsxtrIfinGlossary ....... §5.8; 820, see also \ifglossaryexists & \glsxtrifemptyglossary
\GlsxtrIfinLabelPrefixlist ....... §11.5.7; 563, 821, see also \glsxtrinlabelprefix & \glsxtrprependlabelprefix
\GlsxtrIfInGlossary ....... 820, see also \ifglossaryexists & \glsxtrifemptyglossary
\GlsxtrIfLinkCounterDef ....... §6.2; 318, 821
\GlsxtrIfnextpunc ....... §5.5.4; 248, 782, 797, 822, 866
\GlsxtrIfPeriod ....... §5.5.4; 247, 248, 796, 822
\GlsxtrIfRecordTrigger ....... §11.4; 524, 822, 859, see also \rgls, \Glsxtr--record-count
\GlsxtrIfUnDefinedOrUndefined ....... §5.10; 15, 265, 283, 822, see also \ifglsused & \glsxtrifwasfirstuse
\GlsxtrIfValueInFieldCsvList ....... §5.13; 303, 304, 822, 938
\GlsxtrIfValueInFieldCodeList ....... §5.13; 52, 54, 191, 192, 246, 252, 253, 282, 283, 350, 351, 823, 866
\GlsxtrIfValueInFieldCodeList ....... §5.13; 52, 253, 823
\GlsxtrIfWasGlslike ....... §5.5.4; 252, 253, 823
\GlsxtrIfWasGlslkeAndFirstUse ....... §5.5.4; 252, 283, 823
\GlsxtrIfWasSubsequentOrShort ....... §5.5.4; 253, 823
\GlsxtrIfWasSubsequentUse ....... §5.5.4; 554, 820

980
Index

252, 823
\GlsXtrIfXpFieldEqXpStr §5.15; 306, 823
\glsxtrIgnorableRules 546, 824
\glsxtrindexaliased §5.9.3; 278, 279, 824
\GlsXtrIndexCounterLink §11.5.8; 573, 824
\glsxtrindexseealso §5.9.3; 22, 33, 266, 270, 278, 824, see also \glsxsee
\glsxtrinitwrgloss §5.1.1; 186, 189, 824
\glsxtrinitwrglossbeforefalse 189, 825
\glsxtrinitwrglossbeforetrue 189, 825
\GLSxtrlinefullformat 175, 825
\Glsxtrlinefullformat 175, 825
\Glsxtrlinefullplformat 175, 825
\glsxtrinsertinsidefalse 132, 826
\glsxtrinsertinsidetrue 131, 826
\GlsxtrInternalLocationHyperlink 826
\glsxtrLatinA 547, 548, 826
\glsxtrLatinAA 551, 826
\glsxtrLatinAEligature 550, 827
\glsxtrLatinE 548, 827
\glsxtrLatinEszettSs 550, 827
\glsxtrLatinEszettSz 550, 827
\glsxtrLatinEth 550, 827
\glsxtrLatinH 549, 827
\glsxtrLatinI 549, 827
\glsxtrLatinInsularG 551, 827
\glsxtrLatinK 549, 828
\glsxtrLatinL 828
\glsxtrLatinLslash 551, 828
\glsxtrLatinM 549, 828
\glsxtrLatinN 549, 828
\glsxtrLatinO 549, 828
\glsxtrLatinOELigature 550, 828
\glsxtrLatinOslash 550, 828
\glsxtrLatinP 549, 829
\glsxtrLatinSchwa 551, 829
\glsxtrLatinT 549, 829
\glsxtrLatinThorn 550, 829
\glsxtrLatinWynn 551, 829
\glsxtrLatinX 550, 829
\GlsxtrLetField §3.5; 39, 40, 667, 829
\GlsxtrLetFieldToField §3.5; 39, 40, 830
\GlsxtrLinkCounterName §6.2; 318, 830
\GlsxtrLinkCounterValue §6.2; 318, 320, 830
\GlsxtrLoadResources §11; 23, 24, 34, 38, 60, 386, 394, 395, 399, 508, 510–512, 518, 521, 536, 603, 830, 860, 944, see also resource options
\glsxtrlocalsetgrouptitle §8.6.4; 426, 830
\glsxtrlocationanchor §11.5.6; 557, 830
\GlsxtrLocationField §8.4.2; 424, 425, 831
\GlsxtrLocationRecordCount §11.4; 522–524, 831
\glsxtrlocrangefmt §8.6.3; 424, 425, 831
\GLXxtrlong §4.3; Table 4.1; 53, 209, 655, 661, 831
\Glsxtrlong §4.3; Table 4.1; 53, 208, 234, 334, 656, 661, 831
\GLSxtrlongformat §4.3; Table 4.1; 18, 19, 43, 47, 52, 54, 56, 61, 63, 65, 74, 81, 93, 103, 115, 159, 175, 208, 233, 235, 333, 364, 496, 640, 656, 661, 717, 831, 866, 908, 923
\GLSxtrlongformatgrp §4.3; Table 4.1; 18, 19, 43, 47, 52, 54, 56, 61, 63, 65, 74, 81, 93, 103, 115, 159, 175, 208, 233, 235, 333, 364, 496, 640, 656, 661, 717, 831, 866, 908, 923
\GLSxtrlonghyphen §4.3; Table 4.1; 53, 208, 234, 334, 656, 661, 831
\Glsxtrlonghyphen §4.3; Table 4.1; 53, 208, 234, 334, 656, 661, 831
\Glsxtrlonghyphennoshort §4.3; Table 4.1; 18, 19, 43, 47, 52, 54, 56, 61, 63, 65, 74, 81, 93, 103, 115, 159, 175, 208, 233, 235, 333, 364, 496, 640, 656, 661, 717, 831, 866, 908, 923
\GLSxtrlonghyphennoshortdescsort §4.3; Table 4.1; 18, 19, 43, 47, 52, 54, 56, 61, 63, 65, 74, 81, 93, 103, 115, 159, 175, 208, 233, 235, 333, 364, 496, 640, 656, 661, 717, 831, 866, 908, 923
\GLSxtrlonghyphennoshort §4.3; Table 4.1; 18, 19, 43, 47, 52, 54, 56, 61, 63, 65, 74, 81, 93, 103, 115, 159, 175, 208, 233, 235, 333, 364, 496, 640, 656, 661, 717, 831, 866, 908, 923
Index

\glsxtrmglsWarnAllSkipped \ §7.9.5; 358, 842
\GLSxtrmultientryadjustedname \ §7.14; 372, 842, 843, 844
\Glsxtrmultientryadjustedname \ §7.14; 372, 843, 844
\Glsxtrmultientryadjustedname \ §7.14; 372, 843, 844
\glsxtrmultientryadjustedname \ §7.14; 372, 373, 842, 843, 844
\GLSxtrmultientryadjustednamefmt \ §7.14; 373, 843
\Glsxtrmultientryadjustednamefmt \ §7.14; 373, 843
\Glsxtrmultientryadjustednamefmt \ §7.14; 373, 843
\glsxtrmultientryadjustednameother \ §7.14; 374, 844
\Glsxtrmultientryadjustednameother \ §7.14; 374, 844
\Glsxtrmultientryadjustednameother \ §7.14; 374, 844
\glsxtrmultientryadjustednamepostsep \ §7.14; 373, 844
\glsxtrmultientryadjustednamepresep \ §7.14; 373, 844
\glsxtrmultientryadjustednamesep \ §7.14; 373, 844
\glsxtrmultilastotherindex \ §7.13; 371, 845
\glsxtrmultilist \ §7.13; 370, 845
\glsxtrmultimain \ §7.13; 370, 845
\glsxtrmultimainindex \ §7.13; 371, 845
\glsxtrmultisupplocation \ §11.5.5; 555, 845
\glsxtrmultitotalelements \ §7.13; 371, 845
\glsxtrnameloclink \ §11.5.6; 560, 845
\glsxtrnamereflink \ §5.7; 257, 258, 563, 568, 846, 847
\glsxtrnewgls \ §5.7; 258, 847
\glsxtrnewGLSlike \ §5.7; 257, 844
\glsxtrnewGLSlike \ §5.7; 257, 259, 568, 846
\glsxtrnewglslink \ §5.7; 258, 847
\glsxtrnewnumber \ §2.1; 11, 19, 32, 494, 847, 919, 948
\glsxtrnewrgls \ §5.7; 258, 847
\glsxtrnewrgls \ §5.7; 259, 847
\glsxtrnewrgls \ §5.7; 258, 259, 847
\glsxtrnewsglssymbol \ §2.1; 10, 16, 18, 32, 494, 847, 919, 950
\glsxtrNoGlossaryWarning \ §15; 96, 848
\GlsXtrNoGlsWarningAutoMake \ §15; 596, 848
\GlsXtrNoGlsWarningBuildInfo \ §15; 595, 848
\GlsXtrNoGlsWarningCheckFile \ §15; 595, 848
\GlsXtrNoGlsWarningEmptyMain \ §15; 594, 848
\GlsXtrNoGlsWarningEmptyNotMain \ §15; 594, 848
\GlsXtrNoGlsWarningEmptyStart \ §15; 594, 848
\GlsXtrNoGlsWarningHead \ §15; 594, 848
\GlsXtrNoGlsWarningMisMatch \ §15; 595, 849
\GlsXtrNoGlsWarningNoOut \ §15; 595, 849
\GlsXtrNoGlsWarningTail \ §15; 595, 849
\glsxtrnoidxgroups \ §8.4; 385, 389, 849
\glsxtrnonprintablerules \ §8.4; 542, 849
\glsxtrnopostpunc \ §8.6.2; 415, 422, 423, 781, 849, 860, see also \nopostdesc
\glsxtronlydescname \ §4.5.3.1; 111, 155, 156, 850
\glsxtronlydescsort \ §4.5.3.1; 111, 155, 156, 850
\glsxtronlyname \ §4.5.3.1; 111, 155, 156, 850
\glsxtronlylong \ §4.5.3.1; 149, 161, 164, 850
\glsxtronlyshort \ §4.5.3.1; 45, 134, 148, 149, 161, 164, 850
\glsxtrnewabbrevpresetkeyhook \ §4.1.5; 45, 162, 846
\glsxtrnewgls \ §5.7; 257, 258, 563, 568, 846, 847
\glsxtrnewgls \ §5.7; 258, 846
\glsxtrnewgls \ §5.7; 257, 568, 846
\glsxtrnewglnumber \ §2.1; 11, 19, 32, 494, 847, 919, 948
\glsxtrnewrgls \ §5.7; 259, 847
\glsxtrnewrgls \ §5.7; 257, 259, 847
\glsxtrnewsglssymbol \ §2.1; 10, 16, 18, 32, 494, 847, 919, 950
\glsxtrnewrgls \ §5.7; 258, 847
\glsxtrnewrgls \ §5.7; 258, 847
\glsxtrnewrgls \ §5.7; 259, 847
\glsxtrnewsglssymbol \ §2.1; 10, 16, 18, 32, 494, 847, 919, 950
Index

\GLSxtrp \ §5.4; 232, 850
\Glsxtrp \ §5.4; 232, 851
\glsxtrp \ §5.4; 32, 183, 185, 186, 230, 231, 677, 756, 850, 851, 866
\glsxtrpageref \ §8.1; 377, 416, 851
\glsxtrparen \ §13; 579, 580, 851, 852
\glsxtrpdfentryfmt \ §5.12.2; 296, 800, 851
\Glsxtrpl \ §13; 579, 580, 851, 852
\glsxtrpostabbrvfootnote \ §4.5.2; 160, 852
\glsxtrpostdescabbreviation \ §8.6.2; 422, 852
\glsxtrpostdescacronym \ §8.6.2; 422, 852
\glsxtrpostdesc(category) \ §8.6.2; 421, 422, 599, 701, 852
\glsxtrpostdescgeneral \ §8.6.2; 248, 250, 755, 856
\glsxtrpostdescindex \ §2.1; 12, 856
\glsxtrpostdescnumber \ §2.1; 11, 856
\glsxtrpostdescription \ §8.6.2; 12, 421–423, 428, 853
\glsxtrpostdescsymbol \ §2.1; 10, 853
\glsxtrpostdescterm \ §8.6.2; 422, 853
\glsxtrpostfootnotelongformat 146, 853
\Glxstrophyphenlong 152, 153, 853, 938
\glsxtrphyphenlong \ §8.6.2; 152–154, 853, 854, 938
\Glsxtrphyphenlongpl \ §154, 853
\glsxtrphyphenlongpl \ §153, 854
\Glsxtrphyphenshort \ §150, 151, 854, 938
\glsxtrphyphenshort \ §150, 151, 153, 854, 938
\Glsxtrphyphenshortpl \ §151, 854
\glsxtrphyphenshortpl \ §151, 854
\Glsxtrphyphenshortsequent \ §151, 152, 854, 939
\glsxtrphyphenshortsequent \ §151, 152, 854, 939
\glsxtrpostlink \ §5.5.4; 246–248, 250, 251, 855, 856
\glsxtrpostlinkAddDescOnFirstUse 855
\glsxtrpostlinkAddSymbolDescOnFirstUse \ §5.5.4; 251, 855
\glsxtrpostlink(category) \ §5.5.4; 248, 250, 599, 701, 855
\glsxtrpostlinkendsentence \ §5.5.4; 248, 250, 855
\glsxtrpostlinkhook \ §5.5.4; 246, 251, 755, 855
\glsxtrpostlinkSymbolDescSep \ §5.5.4; 251, 855
\glsxtrpostlocalreset \ §5.10; 280, 856
\glsxtrpostlocalunset \ §5.10; 280, 856
\glsxtrpostlongdescription \ §3.1; 32, 856, 899
\glsxtrpostname(category) \ §8.6.1; 420, 421, 439, 599, 603, 701, 856
\glsxtrpostnamehook \ §8.6.1; 420, 574, 603, 713, 856
\GlsXtrPostNewAbbreviation \ §4.5.3.1; 163, 165, 166, 857
\glsxtrpostreset \ §5.10; 280, 857
\glsxtrpostset \ §5.10; 280, 857
\glsxtrpostuserlongformat \ §11.5.7; 564, 857
\glsxtrpostusershortformat \ §11.5.7; 563, 857
\GlsXtrPostGlossaryStyle \ §8.6; 393, 418, 858, 934
\glsxtrpreglossarystyle \ §8.6.5; 427, 428, 432, 440, 858
\glsxtrprelocation \ §8.6.5; 427, 428, 432, 440, 858
\glsxtrprependlabelprefix \ §11.5.7; 563, 858
\GlsXtrPreToDefaultGlsOpts \ §5.1.1; 186, 187, 858
\glsxtrprovideaccsuppcmd \ 168, 169, 858
\GlsXtrProvideBibTeXFields \ §11.5.2; 536, 785–788, 858
\glsxtrprovidecommand \ §11.5.8; 572, 858

984
Index

\glsxtrprovidestoragekey \ §3.2; 35, 554, 859
\glsxtrrecentanchor \ §11.5.6; 557, 559, 859, 877
\GlsxXtrRecordCount \ §11.4; 523, 524, 859
\GlsxXtrRecordCounter \ §8.4.3.2; 407, 615, 779, 859, 928
\glsxtrrecorddrippergvalue \ §11.4; 524, 528, 529, 822, 859, see also \rgls & --record-count
\GlsxXtrRecordWarning \ §8.5.4
\glsxtrregularfont \ §5.5.2; 60, 65, 183, 236, 240, 255, 706, 859
\GlsxXtrResetLocalBuffer \ §5.10.1; 285, 287, 288, 860, see also \GlsxXtr-UnsetBufferEnableRepeatLocal
\glsxtrresourcecount \ §11; 511, 830, 860
\glsxtrresourcefile \ §11; 23, 24, 510–512, 603, 830, 842, 860, see also resource options & \GlsxXtrLoadResources
\glsxtrresourceinit \ §11; 512, 860
\glsxtrrestoremarkhook \ §5.3.3; 227, 418, 837, 860
\glsxtrrestorepostpunc \ §8.6.2; 423, 860
\glsxtrrevert \ §5.5.4; 133, 171, 794, 798, 860, 862, 873
\glsxtrRevertMarks \ §5.3; 201, 204, 214, 821, 861, see also \glsxtrRevertTocMarks
\glsxtrRevertTocMarks \ §5.3; 201, 204, 214, 821, 861, see also \glsxtrRevertMarks
\glsxtrsavesaveinsert \ §5.5.4; 54, 253, 807, 861
\glsxtrscfoot \ §5.4; 861
\glsxtrsconlydescname \ 113, 156, 861
\glsxtrsconlydessort \ 113, 156, 861
\glsxtrsconlyname \ 112, 156, 861
\glsxtrsconlyrevert \ 155, 862
\glsxtrsconlysuffix \ 113, 155, 862
\glsxtrscrho \ 137, 155, 157, 862
\glsxtrscsuff \ 67–69, 75, 76, 82, 83, 94, 95, 118–121, 137, 155, 157, 171, 654, 862
\glsxtrscuserrevert \ 136, 137, 862
\glsxtrscusersuffix \ 91, 137, 862
\glsxtrseealsolabels \ §5.9.2; 277, 862
\glsxtrseeelist \ §5.13; 299, 863
\glsxtrseeelists \ §5.9.2; 276, 863, see also \glsxtrseeelistsencap & \glsxtrseeelistsdelim
\glsxtrseeelistsdelim \ §5.9.2; 277, 863
\glsxtrseeelistsencap \ §5.9.2; 276, 277, 863
\glsxtrsetactualanchor \ §11.5.6; 557, 863
\GlsxXtrSetActualChar \ §12; 577, 863
\glsxtrsetaliasnoindex \ §5.9.3; 267, 278, 279, 824, 864
\GlsxXtrSetAltModifier \ §5; 184, 309, 326, 842, 864
\glsxtrsetbiblgsaux \ §2.4; 26, 864
\glsxtrsetcategory \ §10; 492, 864
\glsxtrsetcategoryforall \ §10; 492, 864
\glsxtrsetcomplexstyle \ §4.5.3.1; 165, 166, 864
\GlsxXtrSetDefaultGlsOpt \ §5.1.1; 186–190, 278, 864
\glsxtrsetdefaultnumberformat \ §5.1.1; 188, 865
\GlsxXtrSetEncapChar \ §12; 577, 865
\GlsxXtrSetEscChar \ §12; 577, 865
\GlsxXtrSetField \ §3.5; 37–40, 272, 390, 492, 599, 672, 674, 795, 829, 865, 938
\glsxtrsetfieldifexists \ §3.5; 39, 865
\glsxtrsetgrouptitle \ §8.6.4; 34, 389, 426, 607, 865
\glsxtrsetglossarylabel \ §8.3; 378, 866
\GlsxXtrSetLevelChar \ §12; 577, 866
\glsxtrsetlongfirstuse \ §4.3; 52, 866
\GlsxXtrSetPlusModifier \ §5; 184, 866
\glsxtrsetpopts \ §5.4; 231, 677, 851, 866
\glsxtrsetpunctuationmarks \ §5.5.4; 249, 866
\glsxtrsetrecordcountattribute \ §11.4; 524, 866
\GlsxXtrSetStarModifier \ §5; 184, 867
Index

\glsxtrsetupfulldefs \textit{§4.3}; 54, 867
\glsxtrSetWidest \textit{§11.5.3}; 553, 867
\glsxtrSetWidestFallback \textit{§11.5.3}; 554, 867
\GLSxtrshort \textit{§4.3}; Table 4.1; 52, 55, 207, 252, 660, 663, 754, 871, 925
\GLSxtrshortpl \textit{§4.3}; Table 4.1; 52, 660, 663, 754, 871, 925
\glsxtrshort \textit{§4.3}; Table 4.1; 51, 207, 659, 662, 765, 867
\glsxtrshortdesclong \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortdesclongsort \textit{§4.3}; Table 4.1; 51, 207, 659, 662, 765, 867
\glsxtrshortformat \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortformatgrp \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortlongdesclong \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortlongdesclongsort \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortdesclong \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortdesclongsort \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortname \textit{§4.3}; Table 4.1; 18, 19, 43, 47, 51, 54, 55, 61, 63, 66, 74, 82, 93, 103, 115, 159, 177, 183, 185, 201, 204, 207, 235, 240, 247, 252, 334, 364, 631, 632, 659, 662, 765, 867, 871, 911, 924
\glsxtrshortpl \textit{§4.3}; Table 4.1; 52, 55, 207, 252, 660, 663, 754, 871
\glsxtrshortpl \textit{§4.3}; Table 4.1; 52, 55, 207, 252, 660, 663, 754, 871, 925
\Glsxtrshortplformat \textit{§2.5}; 30, 872
\glsxtrshortplformatgrp \textit{§2.5}; 30, 872
\glsxtrshortplformat \textit{§2.5}; 30, 872
\glsxtrshortplformatgrp \textit{§2.5}; 30, 872
\glsxtrshortplformat \textit{§2.5}; 30, 872
\glsxtrshortplformatgrp \textit{§2.5}; 30, 872
\glsxtrshortplformat \textit{§2.5}; 30, 872
\glsxtrshortplformatgrp \textit{§2.5}; 30, 872
\glsxtrshortplformat \textit{§2.5}; 30, 872
\glsxtrshortplformatgrp \textit{§2.5}; 30, 872
\glsxtrshowtargetinner \textit{§2.5}; 30, 872
\glsxtrshowtargetouter \textit{§2.5}; 30, 872
\glsxtrshowtargetsymbolleft \textit{§2.5}; 30, 872
\glsxtrshowtargetsymbolright \textit{§2.5}; 30, 872
\glsxtrsmfont \textit{§2.5}; 873
\glsxtrsmrevert \textit{§2.5}; 873
\glsxtrmsuffix \textit{§2.5}; 873
\glsxtrsmrules \textit{§2.5}; 873
\GlsXtrStandaloneEntryHeadName \textit{§8.5}; 417, 873
\GlsXtrStandaloneEntryHeadOther \textit{§8.5}; 418, 874
\GlsXtrStandaloneEntryName \textit{§8.5}; 256, 415, 874
\GlsXtrStandaloneEntryOther \textit{§8.5}; 415, 874
\GlsXtrStandaloneEntryPdfName \textit{§8.5}; 417, 874
\GlsXtrStandaloneEntryPdfOther \textit{§8.5}; 417, 874
\GlsXtrStandaloneGlossaryType \textit{§8.5}; 414, 874
\GlsXtrStandaloneSubEntryItem \textit{§8.5}; 414, 874
\GlsXtrStartUnsetBuffering \textit{§5.10.1}; 284, 285, 875, see also \GlsXtrStopUnsetBuffering & \GlsXtrDiscardUnsetBuffering
\GlsXtrStopUnsetBuffering \textit{§5.10.1}; 285, 289, 875, see also
Index

\GlsXtrStartUnsetBuffering, \GlsXtrDiscardUnsetBuffering & \GlsXtrForUnsetBufferedList \GLSxtrtitlefullpl \§5.3.3; 223, 723, 879, see also \Glsfmtfullpl \glsxtrheadfull
\glsxtrSubscriptDigitrules ... 545, 875 \GLSxtrtitlefullpl \§5.3.3; 222, 723, 879, see also \Glsfmtfullpl
\GLSxtrsubsequentfmt ... 173, 875 \GLSxtrtitlefullpl \§5.3.3; 222, 723, 879, see also \Glsfmtfullpl
\glsxtrsubsequentfmt ... 61, 173, 255, 875 \GLSxtrheadfullpl
\GLSxtrsubsequentplfmt ... 173, 876 \GLSxtrtitlefullpl \§5.3.3; 222, 723, 879, see also \Glsfmtfullpl
\glsxtrsubsequentplfmt ... 173, 876 \GLSxtrheadfullpl
\glxstrSuperScriptDigitrules 545, 876 \GLSxtrtitlelong ... \§5.3.3; 220, 879
\glxstrsuphypernumber ... \§5.1.2; 195, 555, 876 \GLSxtrtitlelong ... \§5.3.3; 220, 879
\glxstrtagfont ... \§4.4; 58, 498, 877 \GLSxtrtitlelong ... \§5.3.3; 220, 879
\glxstrtaggedlist ... \§5.13; 300, 877 \GLSxtrtitlelong ... \§5.3.3; 221, 880
\glxstrtaggedlistsep ... \§5.13; 300, 877 \GLSxtrtitlename \§5.3.3; 223, 880, see also \Glsfmtname & \GLSxtrheadname
\GLSxtrTheLinkCounter ... \§6.2; 318, 877 \GLSxtrtitlename \§5.3.3; 223, 880, see also \Glsfmtname & \GLSxtrheadname
\GLSxtrtitlenamereflink ... \§11.5.6; 559, 877 \GLSxtrtitlenamereflink ... \§5.3.3; 220, 879
\GLSxtrtitlefirst ... \§5.3.3; 226, 877, see also \GLSxtrtitlefirst & \GLSxtrheadfirst
\GLSxtrtitlefirst ... \§5.3.3; 225, 878, see also \GLSxtrtitlefirst & \GLSxtrheadfirst
\GLSxtrtitlefirst ... \§5.3.3; 225, 878, see also \GLSxtrtitlefirst & \GLSxtrheadfirst
\GLSxtrtitlefirst ... \§5.3.3; 226, 878, see also \GLSxtrtitlefirstpl & \GLSxtrheadfirst
& \GLSxtrheadfirstpl
\GLSxtrtitlefirstpl ... \§5.3.3; 226, 878, see also \GLSxtrtitlefirstpl & \GLSxtrheadfirstpl
\GLSxtrtitlefirstpl ... \§5.3.3; 226, 878, see also \GLSxtrtitlefirstpl & \GLSxtrheadfirstpl
\GLSxtrtitlefull ... \§5.3.3; 222, 722, 878, see also \GLSxtrtitlefull
\GLSxtrtitlefull ... \§5.3.3; 222, 722, 878, see also \GLSxtrtitlefull & \GLSxtrheadfull
\GLSxtrtitlefull ... \§5.3.3; 221, 723, 879, see also \GLSxtrtitlefull & \GLSxtrheadfull
\GLSxtrtitlefull ... \§5.3.3; 221, 723, 879, see also \GLSxtrtitlefull & \GLSxtrheadfull
...

987
Index

\text{Glsfmttext} & \text{Glsxtrheadtext} & \text{GlsXtrUseAbbrStyleSetup} & §4.5.2; 159, 887
\text{Glsxtrtitletext} §5.3.3; 224, 882, see also
\text{Glsfmttext} & \text{Glsxtrheadtext} & \text{Glsxtrusealias} §5.9.2; 277, 887, see also
\text{GlsXtrTotalRecordCount} §11.4; 523, 882
\text{Glsxtrtreechildpredesc} 882
\text{Glsxtrtreepredesc} 883
\text{Glsxtrundefinedaction} §2.4; 16, 883
\text{Glsxtrundefhtag} §2.4; 15, 883
\text{GlsXtrUnknownDialectWarning} §5.12.1; 294, 883
\text{GlsXtrUnsetBufferDisableRepeatLocal} §5.10.1; 286, 883
\text{GlsXtrUnsetBufferEnableRepeatLocal} §5.10.1; 189, 285, 287, 860, 883, see also \text{GlsXtrResetLocalBuffer}
\text{Glsxtrunrsrdto} §8.4.3; 402, 408, 883
\text{Glsxtrunusedformat} §5.9.3; 20, 279, 884
\text{Glsxtrupalpha} 553, 884
\text{Glsxtrupbeta} 553, 884
\text{Glsxtrupchi} 553, 884
\text{Glsxtrupdelta} 553, 884
\text{Glsxtrupdigamma} 553, 884
\text{Glsxtrupepsilon} 553, 884
\text{Glsxtrupeta} 553, 884
\text{Glsxtrupgamma} 553, 885
\text{Glsxtrupiota} 553, 885
\text{Glsxtrupkappa} 553, 885
\text{Glsxtruplambda} 553, 885
\text{Glsxtrupmu} 553, 885
\text{Glsxtrupnu} 553, 885
\text{Glsxtrupomega} 553, 885
\text{Glsxtrupomicron} 553, 885
\text{Glsxtrupphi} 553, 886
\text{Glsxtruppi} 553, 886
\text{Glsxtrupsigmasmall} 553, 886
\text{Glsxtrupsigma} 553, 886
\text{Glsxtruptau} 553, 886
\text{Glsxtruptheta} 553, 886
\text{Glsxtrupupsilon} 553, 886
\text{Glsxtrupupsigmasmall} 553, 887
\text{GlsXtrUseAbbrStyleFmts} §4.5.2; 160, 887
\text{Glsxtrusecursive} §5.9.2; 22, 277, 892, see also

988
\glsxtrusealias, \glsxtruseseealso & \glsxtrseelists \glsxtruseseealso §5.9.2; 22, 277, 892, see also \glsxtrusesee, \glsxtrusealias & \glsxtrseelists \glsxtruseseealsoformat §5.9.2; 33, 267, \hfil \hl \hsize \hyperbf \hyperlink \hyperref \hyperref package 25, 27, 36, 63, 194, 196, 197, 201, 205, 216, 230, 239, 256, 291, 292, 296, 313, 413, 417, 510, 522, 529, 555, 556, 576, 586, 591, 800 \ifcsstring \ifcsundefined \ifglossaryexists ... 671, 894, see also \doifglossarynoexistsord & \glsxtrifemptyglossary \ifglsentryexists ... 703, 894, see also \glsdoifexistsord, \glsdoifexists & \glsdoifnoexists \ifglsfieldcsseq \ifglsfielddefeq \ifglsfieldhaschildren ... 554, 895, see also \GlsXtrIfHasNonZeroChildCount \ifglsfielddesc \ifglsfielddescsuppressed ... 895 \ifglsfieldhasfield 305, 700, 820, 895, see also \glsxtrifhasfield & \ifglsfieldvoid \ifglsfieldhaslong ... 896 \ifglsfieldhasparent ... 292, 554, 896 \ifglsfieldshort ... 255, 896 \ifglsfieldssymbol ... 896 \ifglsindexonlyfirst ... §5.8; 265, 896 \ifGlsLongExtraUseTabular §8.7.2; 445, 748, 896 \ifglssnogroupskip ... 428, 896 \ifglssresetcurrcount ... §6.1; 308, 756, 757, 897 \ifglssused ... 15, 246, 265, 281–283, 823, 897, see also \GlsXtrIfUnusedOrUndefined & \glsxtrifwasfirst & \ifglsxtrinitwrglossbefore ... §5.1.2; 194, 897 \ifglsxtrinsertinside ... 131, 174, 176, 177, 826, 832, 868, 897 \ifGlsXtrPrefixLabelFallbackLast §11.5.7; 563, 857, 897 \ifglsxtrprintglossflatten ... §8.4.3;

Index

\glsxtrusealias, \glsxtruseseealso & \glsxtrseelists \glsxtruseseealso §5.9.2; 22, 277, 892, see also \glsxtrusesee, \glsxtrusealias & \glsxtrseelists \glsxtruseseealsoformat §5.9.2; 33, 267, 276–278, 863, 892 \GlsXtrWarnDeprecatedAbbrStyle §4.5.2; 160, 892 \GlsXtrWarning §13; 579, 892 \glsxtrword ... §10.2.1.1; 174, 496, 497, 893 \glsxtrwordsep ... §10.2.1.1; 149, 151, 496, 497, 893 \glsxtrwordsephyphen ... 149, 151, 893 \glsxtrwrglossaryhook ... 893 \glsxtrwrglossarylocfmt ... §11.5.6; 29, 559, 893 \glsxtrwrglosscountermark §2.5; 29, 893 \glsxtrwrglossmark §2.5; 29, 893 \hfil \hsize \hyperbf \hyperlink \hyperref \hyperref package 25, 27, 36, 63, 194, 196, 197, 201, 205, 216, 230, 239, 256, 291, 292, 296, 313, 413, 417, 510, 522, 529, 555, 556, 576, 586, 591, 800 \ifcsstring \ifcsundefined \ifglossaryexists ... 671, 894, see also \doifglossarynoexistsord & \glsxtrifemptyglossary \ifglsentryexists ... 703, 894, see also \glsdoifexistsord, \glsdoifexists & \glsdoifnoexists \ifglsfieldcsseq \ifglsfielddefeq \ifglsfieldeq \ifglsfieldinvalid \ifglsfieldvoid \ifglsfieldvoid \ifglsfieldhaschildren ... 554, 895, see also \GlsXtrIfHasNonZeroChildCount \ifglsfielddesc \ifglsfielddescsuppressed ... 895 \ifglsfieldhasfield 305, 700, 820, 895, see also \glsxtrifhasfield & \ifglsfieldvoid \ifglsfieldhaslong ... 896 \ifglsfieldhasparent ... 292, 554, 896 \ifglsfieldshort ... 255, 896 \ifglsfieldssymbol ... 896 \ifglsindexonlyfirst ... §5.8; 265, 896 \ifGlsLongExtraUseTabular §8.7.2; 445, 748, 896 \ifglssnogroupskip ... 428, 896 \ifglssresetcurrcount ... §6.1; 308, 756, 757, 897 \ifglssused ... 15, 246, 265, 281–283, 823, 897, see also \GlsXtrIfUnusedOrUndefined & \glsxtrifwasfirst & \ifglsxtrinitwrglossbefore ... §5.1.2; 194, 897 \ifglsxtrinsertinside ... 131, 174, 176, 177, 826, 832, 868, 897 \ifGlsXtrPrefixLabelFallbackLast §11.5.7; 563, 857, 897 \ifglsxtrprintglossflatten ... §8.4.3;

H

hierarchical level 261, 379, 384, 386, 388, 392, 393, 398, 400, 402, 406, 435, 438, 444, 468, 472, 530, 553, 600, 608, 625, 700, 715, 716, 727, 760, 774, 883 h1 ... 241 hsize ... 429 \hyperbf ... 560, 894 \hyperlink ... 317 \hyperref ... 573 hyperref package ... 25, 27, 36, 63, 194, 196, 197, 201, 205, 216, 230, 239, 256, 291, 292, 296, 313, 413, 417, 510, 522, 529, 555, 556, 576, 586, 591, 800

I

\ifcsstring \ifcsundefined ... 305, 819 \ifcsvoid \ifdefempty \ifdefstring \ifndefempty \ifndefstring \ifndef

989
## Index

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>\finlistcs</td>
<td>305, 801</td>
</tr>
<tr>
<td>\ifmglsused</td>
<td>§7.7: 352, 898</td>
</tr>
<tr>
<td>\ifmultiglossaryentryglobal</td>
<td>§7: 325, 898</td>
</tr>
<tr>
<td>\IfTeXParserLib</td>
<td>§11.5.8: 572, 898</td>
</tr>
<tr>
<td>\IfTrackedLanguageFileExists</td>
<td>538</td>
</tr>
<tr>
<td>\ifdef</td>
<td>305</td>
</tr>
<tr>
<td>ignored glossary</td>
<td>256, 259, 375, 376, 382, 383, 405, 435, 525, 601, 673, 715, 726, 727, 736, 818, 820, 894, 900, 918, 926, 927</td>
</tr>
<tr>
<td>ignored location (or record)</td>
<td>261, 513, 522, 524, 601</td>
</tr>
<tr>
<td>\index</td>
<td>261, 501, 502, 574–576, 578</td>
</tr>
<tr>
<td>indexing application</td>
<td>195, 261, 262, 311, 425, 600, 601, 944, 946</td>
</tr>
<tr>
<td>\indexname</td>
<td>898, 947</td>
</tr>
<tr>
<td>\indexspace</td>
<td>430</td>
</tr>
<tr>
<td>inline full form</td>
<td>50, 51, 53, 54, 63, 65, 67, 73, 74, 81, 88, 93, 100, 101, 103, 105–107, 109, 111, 114, 132, 170, 175, 252, 282, 599, 601, 636, 639, 793, 807, 825, 826, 861</td>
</tr>
<tr>
<td>\input</td>
<td>508</td>
</tr>
<tr>
<td>\inputenc package</td>
<td>578</td>
</tr>
<tr>
<td>internal field</td>
<td>35, 37, 264, 303, 600, 602</td>
</tr>
<tr>
<td>internal field (bib2gls)</td>
<td>34, 602, 607</td>
</tr>
<tr>
<td>internal field label</td>
<td>38–40, 135, 231, 244, 245, 291, 293, 295, 301–303, 305, 366, 398, 415, 419, 470, 566, 602, 677, 713, 714,</td>
</tr>
<tr>
<td>\label</td>
<td>27, 378, 624, 866</td>
</tr>
<tr>
<td>\leavevmode</td>
<td>32</td>
</tr>
<tr>
<td>\let</td>
<td>227, 243</td>
</tr>
<tr>
<td>\letabbreviationstyle</td>
<td>§4.5.3: 161, 898</td>
</tr>
<tr>
<td>\letcs</td>
<td>475</td>
</tr>
<tr>
<td>\levelchar</td>
<td>576</td>
</tr>
<tr>
<td>\linewidth</td>
<td>449, 457</td>
</tr>
<tr>
<td>\listcsadd</td>
<td>39, 801</td>
</tr>
<tr>
<td>\listcseadd</td>
<td>39, 801</td>
</tr>
<tr>
<td>\listcsgadd</td>
<td>39, 802</td>
</tr>
<tr>
<td>\listcsxadd</td>
<td>39, 802</td>
</tr>
<tr>
<td>\loadglsentries</td>
<td>508, 899</td>
</tr>
<tr>
<td>\location</td>
<td>188, 195, 355, 513, 555, 601, 602, 614, 616, 729, 731, 752, 884</td>
</tr>
<tr>
<td>location, ignored/invisible</td>
<td>see ignored location (or record)</td>
</tr>
</tbody>
</table>

990
Index


\longnewglossaryentry §3.1; 16, 17, 32, 514, 856, 899
\longnewglossaryentry* §3.1; 32, 386, 899
longtable environment ... 399, 445, 449–458, 460, 461, 468, 469, 472, 645, 647–649, 737–744, 747, 896
longtable package ... 649
lowercase ... 136, 157, 200, 262, 552, 553, 750, 839

M

\mainmatter 187, 188
\MakeAcronymsAbbreviations §4.6; 181, 182, 899
\makefirstuc ... 197–199, 241, 291, 292, 373, 677, 717, 720, 721, 750, 753, 754, 758, 765, 806, 807, 831, 834, 867, 871, 899, 900, 901
\mkglossaries §8; 23–25, 256, 375, 382, 385, 588, 755, 849, 900, 926, 927, 944, 949
\mkglossaries ... 6, 256, 312, 313
\makeindex 6, 17, 20, 23, 25, 28, 195, 196, 261, 574, 601, 900, 926, 944, 947
\makeoffsetglossaries 16, 17, 24, 25, 375, 382, 397, 513, 555, 755, 849, 900, 927, 944, 949
\MakeTextLowercase 200
\MakeTextUppercase 203, 227, 900
\MakeUppercase 149, 150, 152, 227
\markboth 201, 204, 214, 227, 418, 652, 821, 837, 861
\markright 201, 204, 214, 227, 418, 652, 821, 837, 861
\mbox ... 241, 242, 284, 288, 289
\makefirstuc §3.1; 16, 17, 32, 514, 856, 899
\makefirstuc §3.1; 32, 386, 899
\makefirstuc \& \MFUblocker ... 199, 299, 750, 842, 899–901, 937, 947
\makefirstuc \& \MFUblocker ... 199, 750, 901, see also \MFUblocker & \MFUblockadmp
\makefirstuc \& \MFUblockadmp ... 198, 750, 901, see also \MFUblocker & \MFUblockadmp
\MFUsave ... 512, 842, 901
\MFUsaveatend ... 512, 901
\MFUsentencecase ... 197, 198, 241, 299, 470, 750, 851, 900, 901
\MGLS §7.11.1; 342, 357, 364, 901
\MGlS §7.11.1; 363, 901
\Mgls §7.11.1; 340, 342, 357, 363, 902
\mglsAddOptions §7; 325, 902
\mglscurrentcategory §7.5; 348, 354, 902
\mglscurrentlabel §7.5; 348, 358, 902
\mglscurrentlist §7.5; 347, 902
\mglscurrentmainlabel §7.5; 347, 902
\mglscurrentmultilabel §7.5; 347, 903
\mglscurrentoptions §7.5; 347, 903
\mglscurrentprefix §7.5; 341, 342, 348, 903
\mglscurrentsuffix §7.5; 342, 348, 903
\mglscustompostlinkhook §7.6; 349, 357, 903
\mglselementindex §7.5; 341, 903
\mglselementposthook §7.5; 347, 904
\mglselementprehook §7.5; 341, 342, 348, 903
\mglscurrentsuffix ... 903
\mglscustompostlinkhook ... 903
\mglselementindex ... 903
Index

366, 904  \mglselementreset  \textit{§7.10}; 362, 904  \textit{\textbackslash mglsemainpl} \textit{§7.11.1}; 339, 340, 363, 909
\mglselementunset \textit{§7.10}; 362, 904  \textit{\textbackslash mglslname} \textit{§7.11.3}; 365, 909
\mglfield \textit{§7.11.3}; 366, 904, 913  \textit{\textbackslash mglslname} \textit{§7.11.3}; 346, 352, 365, 909
\mglforelements \textit{§7.13}; 370, 371, 904, 905  \textit{\textbackslash MGPls} \textit{§7.11.1}; 364, 909
\mglfortheelements \textit{§7.13}; 371, 904, 905  \textit{\textbackslash MGPls} \textit{§7.11.1}; 364, 910
\Mglsfull \textit{§7.11.2}; 365, 905  \textit{\textbackslash MGlspl} \textit{§7.11.1}; 363, 910
\mglfull \textit{§7.11.2}; 364, 365, 905  \textit{\textbackslash MGlspl} \textit{§7.11.1}; 340, 346, 360, 903, 910
\mglhascategoryprefix \textit{§7.3}; 342, 905  \textit{\textbackslash mglsprefix} \textit{§7.3}; 340, 342, 910
\mglhascategorysuffix \textit{§7.3}; 342, 905  \textit{\textbackslash mglsreset} \textit{§7.7}; 352, 910
\mgliflast  \textit{§7.5}; 349, 905  \textit{\textbackslash mglsresetall} \textit{§7.7}; 353, 910
\mgliflastelementcapscase \textit{§7.6.1}; 351, 905  \textit{\textbackslash mglsseefirstitem} \textit{§7.12}; 301, 369, 758, 910
\mgliflastelementskipped \textit{§7.6.1}; 350, 906  \textit{\textbackslash mglsseeitem} \textit{§7.12}; 301, 369, 758, 911
\mgliflastelementwasfirstuse \textit{§7.6.1}; 350, 906  \textit{\textbackslash mglsSetMain} \textit{§7}; 325, 911
\mgliflastelementwasplural \textit{§7.6.1}; 351, 906  \textit{\textbackslash mglsSetOptions} \textit{§7}; 325, 911
\mgliflastmaincapscase \textit{§7.6.2}; 352, 906  \textit{\textbackslash MGllsshort} \textit{§7.11.2}; 364, 911
\mgliflastmainskipped \textit{§7.6.2}; 351, 906  \textit{\textbackslash mgllsshort} \textit{§7.11.2}; 364, 911
\mgliflastmainwasfirstuse \textit{§7.6.2}; 351, 906  \textit{\textbackslash mgllssuffix} \textit{§7.3}; 341, 342, 910
\mgliflastmainwasplural \textit{§7.6.2}; 352, 907  \textit{\textbackslash MGllssymbol} \textit{§7.11.3}; 365, 366, 912
\mglisfirstuse \textit{§7.5}; 348, 907  \textit{\textbackslash mgllssymbol} \textit{§7.11.3}; 366, 369, 912
\mgllastcategory \textit{§7.6}; 350, 907  \textit{\textbackslash mglsreset} \textit{§7.7}; 352, 912
\mgllastelementlabel \textit{§7.6.1}; 341, 350, 351, 907  \textit{\textbackslash mglsresetall} \textit{§7.7}; 353, 912
\mgllastelementpostlinkhook \textit{§7.6}; 349, 357, 907  \textit{\textbackslash mglsunset} \textit{§7.7}; 352, 912
\mgllastmainlabel \textit{§7.6.2}; 351, 907  \textit{\textbackslash mglsunset} \textit{§7.7}; 353, 912
\mgllastmainpostlinkhook \textit{§7.6}; 349, 357, 907  \textit{\textbackslash mglsunset others} \textit{§7.10}; 362, 912
\mgllastmultilabel \textit{§7.6}; 350, 908  \textit{\textbackslash mglsusecategoryprefix} \textit{§7.3}; 342, 912
\mgllocalreset \textit{§7.7}; 353, 908  \textit{\textbackslash mglsusecategoriesuffix} \textit{§7.3}; 342, 913
\mgllocalunset \textit{§7.7}; 353, 908  \textit{\textbackslash MGPlsusefield} \textit{§7.11.3}; 366, 913
\mgllocalunsetothers \textit{§7.10}; 362, 908  \textit{\textbackslash MGPlsusefield} \textit{§7.11.3}; 366, 913
\Mglslong \textit{§7.11.2}; 365, 908  \textit{\textbackslash MGPlsusefield} \textit{§7.11.3}; 366, 904, 913
\mglslong \textit{§7.11.2}; 364, 365, 908  \textit{\textbackslash MGPlsusefield} \textit{§7.11.3}; 366, 914
\MGLSmainpl \textit{§7.11.1}; 364, 908  \textit{\textbackslash MGPlsmainpl} \textit{§7.11.4}; 368, 914
\MGLSmainpl \textit{§7.11.1}; 364, 908  \textit{\textbackslash MGPlsmainpl} \textit{§7.11.4}; 368, 914
\MGLSmainpl \textit{§7.11.1}; 363, 909  \textit{\textbackslash MGPlspl} \textit{§7.11.4}; 368, 914
\MGLSmainpl \textit{§7.11.1}; 363, 909  \textit{\textbackslash MGPlspl} \textit{§7.11.4}; 368, 914

992
null
Index

\newsym \hspace{1em} §2.4; 18, 919, 945
\newterm \hspace{1em} §2.1; 12, 494, 919, 947
\NoCaseChange \hspace{1em} 203, 216
non-breaking space (-) \hspace{1em} 329, 345, 759
\nopostdesc \hspace{1em} 12, 32, 415, 422, 781, 895, 919,
see also \glsxtrnopostpunc
\Nu \hspace{1em} §11.5.8; 573, 919
\null \hspace{1em} 6
number list \hspace{1em} see location list
\Omicron \hspace{1em} §11.5.8; 573, 919
\omicron \hspace{1em} §11.5.8; 573, 919

\P

page counter \hspace{1em} 25, 27, 261, 264, 313, 317, 523, 529, 530, 555, 559, 877
\pagelistname \hspace{1em} 446, 920
\pagerefname \hspace{1em} 377, 851
\pagemode \hspace{1em} 206
\par \hspace{1em} 752
\parindent \hspace{1em} 465
\part \hspace{1em} 261
\partcounter \hspace{1em} 261
\pdfbookmark \hspace{1em} 442, 788
period (.) \hspace{1em} see full stop (.)
\PGLS \hspace{1em} 369, 920
\Pgls \hspace{1em} 55, 207, 368, 920
\pglsw \hspace{1em} 55, 367, 368, 914, 920, 922
\PGLSfmtlong \hspace{1em} §5.3.2; 209, 920
\Pgsfmtlong \hspace{1em} §5.3.2; 209, 219, 920, 925
\pgsfshort \hspace{1em} §5.3.2; 209, 920
\Pgsfmtlongpl \hspace{1em} §5.3.2; 210, 921
\pysfmtlongpl \hspace{1em} §5.3.2; 210, 921
\PGLSfmtshort \hspace{1em} §5.3.2; 207, 921
\Pgsfmtshort \hspace{1em} §5.3.2; 207, 219, 921, 925
\pgsfmtshort \hspace{1em} §5.3.2; 207, 219, 921, 922
\PGLSfmtshortpl \hspace{1em} §5.3.2; 208, 922
\pysfmtshortpl \hspace{1em} §5.3.2; 208, 922
\PGLSpl \hspace{1em} 369, 922
\Pglsp \hspace{1em} 368, 922
\pglspl \hspace{1em} 368, 914, 915, 923
\PGLSxtrlong \hspace{1em} §4.3.1; 56, 923
\Pgsxtrlong \hspace{1em} §4.3.1; 56, 923
\paxtrlong \hspace{1em} §4.3.1; 55, 923
\PGLSxtrlongpl \hspace{1em} §4.3.1; 56, 923
\Pgsxtrlongpl \hspace{1em} §4.3.1; 56, 923
\paxtrlongpl \hspace{1em} §4.3.1; 56, 924
\PGLSxtrshort \hspace{1em} §4.3.1; 55, 924
\Pgsxtrshort \hspace{1em} §4.3.1; 55, 924
\paxtrshort \hspace{1em} §4.3.1; 55, 924
\PGLSxtrshortpl \hspace{1em} §4.3.1; 55, 924
\Pgsxtrshortpl \hspace{1em} §4.3.1; 55, 924
\paxtrshortpl \hspace{1em} §4.3.1; 55, 924
\PGLSxtrtitlelong \hspace{1em} §5.3.3; 219, 925
\Pgsxtrtitlelong \hspace{1em} §5.3.3; 219, 925
\paxtrtitlelong \hspace{1em} §5.3.3; 219, 925
\PGLSxtrtitleshort \hspace{1em} §5.3.3; 219, 925
\Pgsxtrtitleshort \hspace{1em} §5.3.3; 219, 925
polyglossia package \hspace{1em} 592
post-description hook \hspace{1em} 12, 33, 269, 305, 402,
422, 439, 446, 459, 466, 468, 599, 603,
605, 646, 700, 701, 737, 745, 849, 852,
853, 919
post-link hook \hspace{1em} 51, 54, 63, 90, 101, 104–107,
109–111, 116, 117, 142, 144, 150–153,
160, 165, 166, 183, 185, 230, 231,
234–236, 238, 239, 245–247, 251–254,
279, 281–283, 296, 330, 334, 349, 356,
357, 495, 525, 528, 599, 600, 603, 606,
621, 630, 633, 639, 641, 698, 700, 701,
731–733, 755, 793, 796, 823, 852–856,
938, 939
post-name hook \hspace{1em} 402, 414, 419–421, 446, 459,
501, 599, 603, 604, 608, 701, 856
\preto \hspace{1em} 165
\pretoglossarypreamble \hspace{1em} §8.2; 377, 925
print "unsrt" glossary commands (and
environment) \hspace{1em} 24, 34, 35, 134, 145,
148, 149, 156, 196, 376, 377, 379, 382,
386, 389, 397, 423, 425, 448, 534, 601,
603, 607, 614, 624, 625, 652, 700, 762,
782, 793, 831, 849, 883, 918, 928, 944
print [unsrt|noidx] glossary options \hspace{1em} 377, 624
entrycounter \hspace{1em} 624
flatten \hspace{1em} §8.3; 379, 391, 400–402, 601,
Index

other .................. §8.7.4; 469, 470, 472
par ..................... §8.7.4; 468, 470, 473
rules .................... §8.7.4; 470
\protect .................. 289
\providecommand .......................... §8; 376, 554, 572, 573, 662, 663, 667, 671, 672, 858, 898, 915, 919, 932, 934–937, 939
\provideignoredglossary ..................... §8; 376, 554, 930
\providemultiglossaryentry .................. §7; 324, 372, 930
\ProvidesGlossariesExtraLang .................. §15; 593, 930
Q
\quotechar .................. 576
R
\ref .......................... 377, 756, 851
\refstepcounter .................. 25, 27, 317, 556
\relax .................. 277, 291, 329, 505, 566, 707, 782, 863, 888
relsize package ........... 63, 157, 632, 634, 636, 643, 644
\renewabbreviationstyle .................. §4.5.3; 161, 930, see also \newabbreviationstyle
\renewcommand .................. §4.5.3; 161, 930, see also \newabbreviationstyle
\renewcommand ............. 250, 422, 572, 573, 756, 858
\RequireGlossariesExtraLang .................. 930
resource file ............... 572, 603
resource options ........... 24–26, 372, 419, 535, 536, 554, 561, 563, 603, 625, 696, 780, 794, 935
alias-loc .................. 267, 278, 279
alias .................. 278
category .................. 528
combine-dual-locations .... 570, 571
compound-adjust-name ........ 372, 843
description-case-change .... 500
dual-backlink .................. 563
dual-field .................. 561, 562
dual-prefix .................. 568
dual-type .................. 561
entry-type-aliases ............ 535, 787
field-aliases .................. 536
flatten .................. 379
group-level .................. 391, 396, 465
group .................. 519
interpret-fields ............ 419
label-prefix .................. 568
loc-counters .................. 529
loc-prefix .................. 424, 428
loc-suffix .................. 424
name-case-change .................. 501
post-description-dot ........ 2, 12, 13
prune-xr .................. 275
replicate-fields ............ 419
save-child-count .................. 468, 472, 554, 614, 820, 895
save-index-counter ......... 27, 573, 614
save-locations .............. 25, 35, 440, 461, 513, 602, 607, 625, 944, 948
secondary .................. 379, 390, 391
see .................. 278, 441
seealso .................. 278, 441
selection .................. 22, 33, 260, 266, 267, 274, 278, 279, 514, 524, 560, 696
set-widest .................. 435, 447, 463, 553, 867
sort-field .................. 34
sort-label-list ............... 299
sort-replace .................. 535
sort-rule .................. 521, 536, 538
sort .................. 25, 378, 514, 536, 944
src .................. 17, 514, 554
supplemental-locations .... 195, 555, 560
symbol-sort-fallback ........ 410
trigger-type .................. 525, 530
type .................. 518
resource set .................. 554, 603, 780, 860
\RestoreAcronyms §4.6; 181, 182, 899, 930
\rGLS §11.4; 199, 259, 526, 527, 847, 931, see also \gsxtrifrecordtrigger, \rGLSformat & --record-count
\rGls §11.4; 200, 258, 526, 527, 847, 931, see also \gsxtrifrecordtrigger, \rGlsformat & --record-count

996
Index

\rgls §11.4; 56, 183, 200, 258, 524, 525, 527, 531, 533, 799, 847, 931, see also
\glsxtrifrecordtrigger,
\rglsformat & --record-count
\rGLSformat §11.4; 526, 931
\rglsformat §11.4; 526, 931
\rglsformat §11.4; 525, 528, 931
\rglspl §11.4; 259, 526, 527, 847, 931, 932, see also \glsxtrifrecordtrigger,
\rglsplformat & --record-count
\rGlspl §11.4; 258, 526, 527, 847, 932, see also \glsxtrifrecordtrigger,
\rGlsplformat & --record-count
\rglspl §11.4; 258, 525, 527, 847, 932, see also \glsxtrifrecordtrigger,
\rglsplformat & --record-count
\rGLSplformat §11.4; 526, 932
\rglsplformat §11.4; 526, 932
\rglsplformat §11.4; 525, 932
\Rho §11.5.8; 573, 932
\Roman §14.2
\roman §14.2

S

sample.tex §14.2; 582
sample-abbr-styles.tex §14.2; 582
sample-abbrv.tex §14.2; 583
sample-accesssupp.tex §14.2; 590
sample-acronym.tex §14.2; 583
sample-acronym-desc.tex §14.2; 584
sample-alias.tex §14.2; 588
sample-altmodifier.tex §14.2; 587
sample-alttree.tex §14.2; 437, 588
sample-alttree-marginpar.tex §14.2; 437, 589
sample-alttree-sym.tex §14.2; 437, 589
sample-autoindex.tex §14.2; 586
sample-autoindex-hyp.tex §14.2; 586
sample-crossref.tex §14.2; 584
sample-entrycount.tex §14.2; 586
sample-external.tex §14.2; 502, 588
sample-fmt.tex §14.2; 588
sample-footnote.tex §14.2; 588
sample-header.tex §14.2; 585
sample-indexhook.tex §14.2; 584
sample-initialisms.tex §14.2; 585
sample-linkcount.tex §14.2; 587
sample-mixed-abbrv-styles.tex §14.2; 585
sample-mixedsort.tex §14.2; 375, 588
sample-mixture.tex §14.2; 583
sample-name-font.tex §14.2; 583
sample-nested.tex §14.2; 586
sample-onelink.tex §14.2; 587
sample-onthefly.tex §14.2; 589
sample-onthefly-utf8.tex §14.2; 589
sample-onthefly-xetex.tex §14.2; 589
sample-pages.tex §14.2; 424, 587
sample-postdot.tex §14.2; 585
sample-postlink.tex §14.2; 585
sample-prefix.tex §14.2; 590
sample-suppl.tex §14.2; 590
sample-suppl-hyp.tex §14.2; 591
sample-suppl-main.tex §14.2; 590
sample-suppl-main-hyp.tex §14.2; 590
sample-undef.tex §14.2; 584
sample-unitentrycount.tex §14.2; 587
\section §14.2
section counter 313, 523, 555, 557, 558
\seealso §33, 267, 278, 605, 609, 824, 932
\seealso §266, 609, 757, 933
sentence case 253, 500, 677, 698, 717, 720, 753, 754, 765, 806, 807, 831, 834, 867, 871, 899, 901, 905, 906
\setabbreviationstyle §4.5; 7, 41, 59, 170, 916, 917, 933
\setacronymystyle \setacronymystyle \setacronymystyle \setacronymystyle 43, 229, 933
\setentrycounter \setentrycounter \setentrycounter \setentrycounter 556, 933
\setglossarypreamble 377, 933
\setglossarystyle 393, 418, 645, 858, 933
\setglossaries §2; 9, 428, 934
\setupglossaries §5.1.1; 190, 934, see also
\setupglslink

997
Index

\setupglslink \§5.1.1; 190, 934, see also \setupglsadd
\so 242, 243
soul package 236, 241, 242, 284
standalone entry 603, 918
\stepcounter 317, 824
\string 512, 521, 535, 539, 541, 575, 698, 728, 935
\subglossentry 398–400, 414, 472, 934
subsequent use 604
supertabular environment 650
\symbolname 446, 469, 934
\tabcolsep 449, 452, 454, 457
table environment 27
\tableofcontents 203, 215
tabular environment 383, 399, 402, 428, 444, 445, 472, 645, 647–649, 668, 672, 702, 748, 752, 896, 920, 934
\Tamath \§11.5.8; 573, 934
texindy 574
texorpdfstring 205, 216, 418
\textbf 430, 560
textcase package 203
\textlarger 158, 873
\textsc 61, 136, 156, 157, 632–634, 636, 641, 642, 765, 802, 861
\textsmalleer 63, 156, 157, 230, 632, 634, 636, 643, 644, 802, 873
\textsubscript 475
\textsuperscript 475
\textup 158, 798
\the\langle\counter\rangle 196, 313, 318, 407, 408, 522, 529, 556
theglossary environment 3, 399, 403, 404, 675, 755, 929, 940, 948
\theglossaryentry 935
\theH\langle\counter\rangle 196, 313, 409, 522, 529, 557, 618
theindex environment 576
\thepage 522, 524
title case 200, 291, 372–374, 500, 665, 710, 802, 843
tracklang package 293, 294, 514, 538, 540, 593, 594, 805
translator package 3, 950
U
\u 512, 521, 535, 728, 935, see also \glshex
\ul 236
\underline 58
\unskip 20, 32, 279
\Upalpha \§11.5.8; 553, 573, 935
\upalpha 551–553
\Upbeta \§11.5.8; 573, 935
\Upchi \§11.5.8; 573, 935
\Upepsilon \§11.5.8; 573, 936
\Upeta \§11.5.8; 573, 936
upgreek package 551, 552, 573, 935–937
\Upiota \§11.5.8; 573, 936
\Upkappa \§11.5.8; 573, 936
\Upmu \§11.5.8; 573, 936
\Upnu \§11.5.8; 573, 936
\upomicron \§11.5.8; 573, 936
\Uprho \§11.5.8; 573, 937
\Uptau \§11.5.8; 573, 937
\Upzeta \§11.5.8; 573, 937
UTF-8 184, 197, 198, 249, 427, 438, 521, 578, 604, 782, 866
W
whatsit 185, 194, 264, 290, 604
wrglossary counter \§2.4; 27, 29, 558, 559, 614, 893, 942, 943
\writemultiglossentry \§7.13; 371, 937

998
### Index

<table>
<thead>
<tr>
<th>Command / Package</th>
<th>Description</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>\xcapitalisefmtwords</td>
<td>................. 500, 937</td>
<td></td>
</tr>
<tr>
<td>xfor package</td>
<td>......................... 301, 371</td>
<td></td>
</tr>
<tr>
<td>\xglssetwidest</td>
<td>$8.6.5.4; 434, 937</td>
<td></td>
</tr>
<tr>
<td>\xglsupdatewidest</td>
<td>$8.6.5.4; 434, 938</td>
<td></td>
</tr>
<tr>
<td>\xGlsXtrIfValueInFieldCsvList</td>
<td>$5.13; 304, 938</td>
<td></td>
</tr>
<tr>
<td>\xGlsXtrSetField</td>
<td>$3.5; 39, 40, 938</td>
<td></td>
</tr>
<tr>
<td>\xifinlistcs</td>
<td>......................... 305, 802</td>
<td></td>
</tr>
<tr>
<td>xindy</td>
<td>6, 17, 20, 23, 25, 28, 195, 196, 261, 574, 601, 610, 900, 926, 944, 950</td>
<td></td>
</tr>
<tr>
<td>\xpglsxtrpostabbrvfootnote</td>
<td>$4.5.2; 160, 938</td>
<td></td>
</tr>
<tr>
<td>\xpglsxtrposthyphenlong</td>
<td>152, 938</td>
<td></td>
</tr>
<tr>
<td>\xpglsxtrposthyphenshort</td>
<td>150, 938</td>
<td></td>
</tr>
<tr>
<td>\xpglsxtrposthyphensubsequent</td>
<td>151, 938</td>
<td></td>
</tr>
<tr>
<td>\xindex</td>
<td>......................... 601</td>
<td></td>
</tr>
<tr>
<td>\Zeta</td>
<td>$11.5.8; 573, 939</td>
<td></td>
</tr>
</tbody>
</table>