

The Biblatex Package

Programmable Bibliographies and Citations

Philipp Lehman Version 3.0

(with Philip Kime, Audrey Boruvka 20/04/2015

and Joseph Wright)

Biblatex 宏包说明文档摘译

胡振震¹

2016-02-08

译者按: 之所以摘译 biblatex, 一是出于对 keep texing 的兴趣, 二是考虑到中文资料里总体介绍 latex 的资料其实较多, 反而是在一些专项部分中文资料较少, 看宏包英文说明当然没有问题, 但有的文档内容长达几百页, 一般用户真心没有精力去看, 即便是找一些自己需要的功能也比较麻烦, 所以考虑对参考文献的 biblatex 宏包文档进行摘译, 是希望能在这一方面有所贡献。应该说, biblatex 宏包的很多功能比如文献表划分, 文献集, 样式定制, 动态数据处理等, 在科技论文或书籍写作特别是对参考文献著录和标注有特殊要求的情况下是很有用的。

总之, 本文总体任务是完成 biblatex 宏包说明文档关键内容的摘译, 希望能对有深度定制参考文献要求的用户有所帮助, 当然也希望能使中文 latex 资料库更为全面和深入。需要说明的是, 限于作者水平, 其中难免存在一些错误和理解不到位的地方, 欢迎批评指正, 欢迎 @译者邮箱。最后感谢 CTEX 和 Latexstudio 论坛, 感谢论坛上各位作者关于 biblatex 参考文献方面的工作分享和经验介绍。

1	引言	1
1.1	关于 biblatex	1
1.2	许可	2
1.3	反馈	2
1.4	致谢	2
1.5	前提与必备	2
1.5.1	必须资源	2
1.5.2	推荐包	4
1.5.3	兼容的包	4
1.5.4	不兼容的包	4
1.5.5	biber/biblatex 兼容性	6
2	数据库指南	6
2.1	条目类型	7
2.1.1	常规类型	7
2.1.2	类型别名	12

¹Email:hzzmail@163.com

2.1.3	不支持的条目类型	12
2.2	条目域 Entry Fields	13
2.2.1	数据类型 Data Types	13
2.2.2	数据域 Data Fields	16
2.2.3	特殊域 Special Fields	28
2.2.4	自定义域 Custom Fields	33
2.2.5	域的别名 Field Aliases	34
2.3	使用注意事项 Usage Notes	35
2.3.1	@inbook条目类型 The Entry Type @inbook	36
2.3.2	缺失和可忽略数据 Missing and Omissible Data	36
2.3.3	集体作者和编者 Corporate Authors and Editors	37
2.3.4	原样输出列表 Literal Lists	37
2.3.5	题名 Titles	38
2.3.6	编辑角色 Editorial Roles	39
2.3.7	出版物和期刊系列 Publication and Journal Series	40
2.3.8	日期格式 Date Specifications	41
2.3.9	月份和期刊季度号 Months and Journal Issues	42
2.3.10	标记页码 Pagination	42
2.4	提示与警告 Hints and Caveats	43
2.4.1	交叉引用 Cross-referencing	43
2.4.2	能力问题 Capacity Issues	45
2.4.3	排序和编码问题 Sorting and Encoding Issues	45
2.4.4	编纂者脚本 Editors and Compiler Scripts	51
3	用户使用手册	51
3.1	宏包选项	51
3.1.1	加载选项	51
3.1.2	导言区选项	52
3.1.3	条目选项 Entry Options	70
3.1.4	遗留选项 Legacy Options	73
3.2	全局定制 Global Customization	73
3.2.1	配置文件 Configuration File	73
3.2.2	设置包选项 Setting Package Options	73
3.3	标准样式 Standard Styles	73
3.3.1	标注样式 Citation Styles	73
3.3.2	参考文献样式 Bibliography Styles	78
3.4	相关条目 Related Entries	80
3.5	排序选项 Sorting Options	82
3.6	参考文献命令 Bibliography Commands	83
3.6.1	文献数据源 Resources	83
3.6.2	参考文献表 The Bibliography	84
3.6.3	缩略表 The List of Shorthands	88
3.6.4	各类参考文献表 Bibliography Lists	88

3.6.5	参考文献分节 Bibliography Sections	91
3.6.6	参考文献分部 Bibliography Segments	91
3.6.7	参考文献分类 Bibliography Categories	92
3.6.8	参考文献标题和环境 Bibliography Headings and Environments	92
3.6.9	参考文献注释 Bibliography Notes	95
3.6.10	参考文献过滤和检查 Bibliography Filters and Checks	95
3.6.11	参考文献表内容 Bibliography Contexts	97
3.6.12	动态条目集 Dynamic Entry Sets	98
3.7	引用命令 Citation Commands	98
3.7.1	标准命令 Standard Commands	99
3.7.2	样式相关命令 Style-specific Commands	99
3.7.3	有限标注表 Qualified Citation Lists	100
3.7.4	与样式无关的命令 Style-independent Commands	102
3.7.5	文本命令 Text Commands	103
3.7.6	特殊命令 Special Commands	104
3.7.7	底层命令 Low-level Commands	107
3.7.8	其它命令 Miscellaneous Commands	107
3.7.9	与natbib兼容的命令 natbib Compatibility Commands	108
3.7.10	类似mcite给的标注命令 mcite-like Citation Commands	109
3.8	本地化命令 Localization Commands	110
3.9	格式命令 Formatting Commands	112
3.9.1	一般命令和钩子 Generic Commands and Hooks	112
3.9.2	语言相关命令 Language-specific Commands	116
3.9.3	尺寸和计数器 Lengths and Counters	118
3.9.4	多用途命令 All-purpose Commands	119
3.10	语言注意点 Language notes	120
3.10.1	美语 American	120
3.10.2	西班牙语 Spanish	121
3.10.3	希腊语 Greek	121
3.10.4	俄语 Russian	122
3.11	用法注意点 Usage Notes	122
3.11.1	概述 Overview	122
3.11.2	辅助文件 Auxiliary Files	123
3.11.3	多个参考文献表 Multiple Bibliographies	124
3.11.4	子参考文献表 Subdivided Bibliographies	127
3.11.5	条目集 Entry Sets	130
3.11.6	数据容器 Data Containers	131
3.11.7	电子出版信息 Electronic Publishing Information	132
3.11.8	额外的摘要和注释 External Abstracts and Annotations	134
3.12	提示与警告 Hints and Caveats	135
3.12.1	与KOMAScript 类共用的方法 Usage with KOMAScript Classes	135
3.12.2	与Memoir 类共用的方法 Usage with the Memoir Class	135

3.12.3	标注中的页码 Page Numbers in Citations	136
3.12.4	姓名组成部分及其间距 Name Parts and Name Spacing	138
3.12.5	参考文献筛选器和标注标签 Bibliography Filters and Citation Labels	139
3.12.6	参考文献标题中的活动字符 Active Characters in Bibliogra- phy Headings	140
3.12.7	在参考文献分节和分部中的编组 Grouping in Reference Sec- tions and Segments	140
4	样式作者指南	140
4.1	概述 Overview	141
4.2	参考文献样式 Bibliography Styles	144
4.2.1	参考文献样式文件 Bibliography Style Files	144
4.2.2	参考文献环境 Bibliography Environments	146
4.2.3	参考文献驱动 Bibliography Drivers	147
4.2.4	特殊域 Special Fields	149
4.3	标注样式 Citation Styles	160
4.3.1	标注样式文件 Citation Style Files	160
4.3.2	特殊域 Special Fields	162
4.4	数据接口 Data Interface	163
4.4.1	数据命令 Data Commands	163
4.4.2	格式化指令 Formatting Directives	168
4.5	自定义 Customization	171
4.5.1	关联条目 Related Entries	171
4.5.2	数据动态修改 Dynamic Modification of Data	172
4.5.3	数据模型规范 Data Model Specification	183
4.5.4	标签 Labels	188
4.5.5	排序 Sorting	195
4.5.6	参考文献表过滤器 Bibliography List Filters	200
4.5.7	控制姓名首字母生成 Controlling Name Initials Generation	201
4.5.8	排序微调 Fine Tuning Sorting	202
4.5.9	特殊域 Special Fields	203
4.5.10	数据继承 Data Inheritance (crossref)	205
4.6	辅助命令 Auxiliary Commands	207
4.6.1	数据命令 Data Commands	207
4.6.2	独立判断命令 Stand-alone Tests	209
4.6.3	使用\ifboolexpr和\ifthenelse的判断 Tests with \ifboolexpr and \ifthenelse	219
4.6.4	综合命令 Miscellaneous Commands	220
4.7	标点 Punctuation	230
4.7.1	块和单元标点 Block and Unit Punctuation	230
4.7.2	标点判断 Punctuation Tests	232
4.7.3	添加标点 Adding Punctuation	232

4.7.4	添加空格 Adding Whitespace	233
4.7.5	标点设置和大写 Configuring Punctuation and Capitalization	234
4.7.6	修正标点追踪 Correcting Punctuation Tracking	236
4.8	本地化字符串 Localization Strings	237
4.9	本地化模块 Localization Modules	239
4.9.1	本地化命令 Localization Commands	239
4.9.2	本地化关键词 Localization Keys	242
4.10	格式化命令 Formatting Commands	254
4.10.1	用户可定义的命令和钩子 User-definable Commands and Hooks	254
4.10.2	不同语言的命令 Language-specific Commands	258
4.10.3	用户可定义的尺寸和计数器 User-definable Lengths and Counters	259
4.10.4	辅助命令和钩子 Auxiliary Commands and Hooks	260
4.10.5	辅助长度计数器和其它功能 Auxiliary Lengths, Counters, and Other Features	263
4.10.6	一般目的的钩子 General Purpose Hooks	265
4.11	提示与警告 Hints and Caveats	268
4.11.1	条目集 Entry Sets	268
4.11.2	电子出版信息 Electronic Publishing Information	269
4.11.3	额外的摘要和注释 External Abstracts and Annotations	270
4.11.4	消除姓名歧义 Name Disambiguation	270
4.11.5	浮动体和TOC/LOT/LOF中的追踪器 Trackers in Floats and TOC/LOT/LOF	275
4.11.6	混合编程接口 Mixing Programming Interfaces	275
4.11.7	使用标点追踪 Using the Punctuation Tracker	277
4.11.8	定制本地化模型 Custom Localization Modules	281
4.11.9	编组 Grouping	283
4.11.10	姓名内的间距 Namespaces	283
附录		284
A 默认的驱动源映射		284
A.1	bibtex	284
A.2	endnotexml	285
A.3	ris	288
A.4	zoterordfxml	290
B 默认的继承设置		293
C 默认的排序方式		294
C.1	Alphabetic 1	294
C.2	Alphabetic 2	295
C.3	Chronological	295
D 选项集合		295

E 更新历史	297
1 Biber/Biblatex compatibility matrix	7
2 Supported Languages	30
3 Date Specifications	42
4 Capacity of bibtex8	46
5 Disambiguation counters	70
6 mcite-like commands	110
7 mcite-like syntax	111
8 Date Interface	157
9 Field types for \nosort	203
10 \mkcomprange setup	225

1 引言

这是关于 biblatex 包的语法文档,使用范例文档参考文档²。快速开始,请浏览 §§ 1.1, 2.1, 2.2, 2.3, 3.1, 3.3, 3.6, 3.7, 3.11 节。

译文文档说明: 译文文档直接在原 biblatex 说明文档基础上进行修改, biblatex 说明文档除了使用了一些基本宏包外主要使用类和包文件包括: scrartcl.cls, ltxdockit.cls, ltxdockit.cfg, ltxdockit.def, btxdockit.sty, ltxdockit.sty。需要修改一些默认设置和命令可在这些文件中查找。

1.1 关于 biblatex

biblatex 包提供了一套与 latex 配合使用的高级参考文献工具。该包是 latex 提供的参考文献功能的完整再实现。该包默认使用自定义后端 biber, 用于处理 bibtex 格式文件、执行排序和生成标签 (以及其它更多工作)。参考文献的格式化完全由 tex 宏包控制。只要具备一定的 latex 知识就足以设计新的参考文献著录样式和标注样式。

biblatex 也支持参考文献表细分, 支持在一个文档内包含多个参考文献表, 支持生成参考文献局部信息表比如多个不同域的缩略词。参考文献表可以根据主题进行分块或者分段。与参考文献著录样式类似, 所有的标注 (引用) 命令也可以自由定义。

提供的功能还包括: 文献数据的 Unicode 支持、自定义排序、不同排序方式的多参考文献表、自定义标签和动态数据修改。biber/biblatex 的版本兼容性见 § 1.5.5 节。该包可完全实现本地化, 可与 babel 或 polyglossia 包配合使用, 该包支持的语言详见表 table 2。

1.2 许可

Copyright © 2006–2012 Philipp Lehman, 2012–2013 Philip Kime, Audrey Boruvka, Joseph Wright. Permission is granted to copy, distribute and/or modify this software under the terms of the LaTeX Project Public License, version 1.3.³

²<http://www.ctan.org/tex-archive/macros/latex/contrib/biblatex/doc/examples>

³<http://www.ctan.org/tex-archive/macros/latex/base/lppl.txt>

1.3 反馈

请使用 `github` 的 `biblatex` 项目页报告 bug 和提交所需功能⁴。若寻求帮助请在 `comp.text.tex` 新闻组或 `tex-latex` 讨论区提交问题⁵。

1.4 致谢

The language modules of this package are made possible thanks to the following contributors: Augusto Ritter Stoffel, Mateus Araújo (Brazilian); Sebastià Vila-Marta (Catalan); Ivo Pletikosić (Croatian); Michal Hoftich (Czech); Jonas Nyrup (Danish); Johannes Wilm (Danish/Norwegian); Alexander van Loon, Pieter Belmans, Hendrik Maryns (Dutch); Hannu Väisänen, Janne Kujanpää (Finnish); Denis Bitouzé (French); Apostolos Syropoulos, Prokopis (Greek); Baldur Kristinsson (Icelandic); Enrico Gregorio, Andrea Marchitelli (Italian); Håkon Malmedal (Norwegian); Anastasia Kandulina, Yuriy Chernyshov (Polish); José Carlos Santos (Portuguese); Oleg Domanov (Russian); Tea Tušar and Bogdan Filipič (Slovene); Ignacio Fernández Galván (Spanish); Per Starbäck, Carl-Gustav Werner, Filip Åsblom (Swedish).

1.5 前提与必备

本节介绍所需资源和兼容性问题。

1.5.1 必须资源

如下资源是必须的，否则 `biblatex` 无法正常工作

- `e-TeX` `Biblatex` 宏包需要 `e-TeX`。
- `Biber` `Biber` 是 `Biblatex` 的默认后端，只需要一个后端 `BibTeX` 或者 `Biber`。
- `BibTeX` `Biblatex` 可以使用传统的 `BibTeX` 作为数据后端。
- `etoolbox` 自动加载，提供 `Biblatex` 所需的通用编程工具。
- `kvoptions` 自动加载，用于内部选项处理。
- `logreq` 自动加载，提供了向辅助日志文件写入机器可读信息的前端。

除了上述资源，`Biblatex` 还需要标准 `LaTeX` 宏包 `keyval`、`ifthen` 和 `url`。这些包包含在所有通用 `TeX` 发行版中，也自动进行加载。

1.5.2 推荐包

The packages listed in this section are not required for `Biblatex` to function, but they provide recommended additional functions or enhance existing features. The package loading order does not matter.

⁴<http://github.com/plk/biblatex>

⁵<http://tex.stackexchange.com/questions/tagged/biblatex>

- `babel/polyglossia` The `babel` and `polyglossia` packages provides the core architecture for multilingual typesetting. If you are writing in a language other than American English, using one of these packages is strongly recommended. If loaded, Biblatex package will detect `babel` or `polyglossia` automatically.
- `csquotes` If this package is available, Biblatex will use its language sensitive quotation facilities to enclose certain titles in quotation marks. If not, Biblatex uses quotes suitable for American English as a fallback. When writing in any other language, loading `csquotes` is strongly recommended.⁶
- `xpatch` The `xpatch` package extends the patching commands of `etoolbox` to Biblatex bibliography macros, drivers and formatting directives.⁷

1.5.3 兼容的包

The Biblatex package provides dedicated compatibility code for the classes and packages listed in this section.

- `hyperref` The `hyperref` package transforms citations into hyperlinks. See the `hyperref` and `backref` package options in § 3.1.2.1 for further details. When using the `hyperref` package, it is preferable to load it after Biblatex.
- `showkeys` The `showkeys` package prints the internal keys of, among other things, citations in the text and items in the bibliography. The package loading order does not matter.
- `memoir` When using the `memoir` class, the default bibliography headings are adapted such that they blend well with the default layout of this class. See § 3.12.2 for further usage hints.
- KOMA-Script When using any of the `scrartcl`, `scrbook`, or `scrreprt` classes, the default bibliography headings are adapted such that they blend with the default layout of these classes. See § 3.12.1 for further usage hints.

1.5.4 不兼容的包

The packages listed in this section are not compatible with Biblatex. Since it reimplements the bibliographic facilities of LaTeX from the ground up, Biblatex naturally conflicts with all packages modifying the same facilities. This is not specific to Biblatex. Some of the packages listed below are also incompatible with each other for the same reason.

- `babelbib` The `babelbib` package provides support for multilingual bibliographies. This is a standard feature of Biblatex. Use the `langid` field and the package option `autolang` for similar functionality. Note that Biblatex automatically adjusts to the main document language if `babel` or `polyglossia` is loaded. You only need the above mentioned features if you want to switch languages on a per-entry basis within the bibliography. See §§ 2.2.3 and 3.1.2.1 for details. Also see § 3.8.

⁶<http://www.ctan.org/tex-archive/macros/latex/contrib/csquotes/>

⁷<http://www.ctan.org/tex-archive/macros/latex/contrib/xpatch/>

- backref** The `backref` package creates back references in the bibliography. See the package options `hyperref` and `backref` in § 3.1.2.1 for comparable functionality.
- bibtopic** The `bibtopic` package provides support for bibliographies subdivided by topic, type, or other criteria. For bibliographies subdivided by topic, see the category feature in § 3.6.7 and the corresponding filters in § 3.6.2. Alternatively, you may use the `keywords` field in conjunction with the `keyword` and `notkeyword` filters for comparable functionality, see §§ 2.2.3 and 3.6.2 for details. For bibliographies subdivided by type, use the `type` and `nottype` filters. Also see § 3.11.4 for examples.
- bibunits** The `bibunits` package provides support for multiple partial (e. g., per chapter) bibliographies. See `chapterbib`.
- chapterbib** The `chapterbib` package provides support for multiple partial bibliographies. Use the `refsection` environment and the `section` filter for comparable functionality. Alternatively, you might also want to use the `refsegment` environment and the `segment` filter. See §§ 3.6.5, 3.6.6, 3.6.2 for details. Also see § 3.11.3 for examples.
- cite** The `cite` package automatically sorts numeric citations and can compress a list of consecutive numbers to a range. It also makes the punctuation used in citations configurable. For sorted and compressed numeric citations, see the `sortcites` package option in § 3.1.2.1 and the `numeric-comp` citation style in § 3.3.1. For configurable punctuation, see § 3.9.
- citeref** Another package for creating back references in the bibliography. See `backref`.
- inlinebib** The `inlinebib` package is designed for traditional citations given in footnotes. For comparable functionality, see the verbose citation styles in § 3.3.1.
- jurabib** Originally designed for citations in law studies and (mostly German) judicial documents, the `jurabib` package also provides features aimed at users in the humanities. In terms of the features provided, there are some similarities between `jurabib` and `Biblatex` but the approaches taken by both packages are quite different. Since both `jurabib` and `Biblatex` are full-featured packages, the list of similarities and differences is too long to be discussed here.
- mcite** The `mcite` package provides support for grouped citations, i. e., multiple items can be cited as a single reference and listed as a single block in the bibliography. The citation groups are defined as the items are cited. This only works with unsorted bibliographies. The `Biblatex` package also supports grouped citations, which are called ‘entry sets’ or ‘reference sets’ in this manual. See §§ 3.11.5, 3.6.12, 3.7.10 for details.
- mciteplus** A significantly enhanced reimplementation of the `mcite` package which supports grouping in sorted bibliographies. See `mcite`.
- multibib** The `multibib` package provides support for bibliographies subdivided by topic or other criteria. See `bibtopic`.

Biber version	Biblatex version
1.9	2.9
1.8	2.8
1.7	2.7
1.6	2.6
1.5	2.5
1.4	2.4
1.3	2.3
1.2	2.1, 2.2
1.1	2.1
1.0	2.0
0.9.9	1.7x
0.9.8	1.7x
0.9.7	1.7x
0.9.6	1.7x
0.9.5	1.6x
0.9.4	1.5x
0.9.3	1.5x
0.9.2	1.4x
0.9.1	1.4x
0.9	1.4x

Table 1: Biber/Biblatex compatibility matrix

- natbib** The `natbib` package supports numeric and author-year citation schemes, incorporating sorting and compression code found in the `cite` package. It also provides additional citation commands and several configuration options. See the `numeric` and `author-year` citation styles and their variants in § 3.3.1, the `sortcites` package option in § 3.1.2.1, the citation commands in § 3.7, and the facilities discussed in §§ 3.6.8, 3.6.9, 3.9 for comparable functionality. Also see § 3.7.9.
- splitbib** The `splitbib` package provides support for bibliographies subdivided by topic. See `bibtopic`.
- titlesec** The `titlesec` package redefines user-level document division commands such as `\chapter` or `\section`. This approach is not compatible with internal command changes applied by the Biblatex `refsection` and `refsegment` option settings described in § 3.1.2.1.
- ucs** The `ucs` package provides support for UTF-8 encoded input. Either use `inputenc`'s standard `utf8` module or a Unicode enabled engine such as XeTeX or LuaTeX instead.

1.5.5 biber/biblatex 兼容性

Biber versions are closely coupled with Biblatex versions. You need to have the right combination of the two. Biber will warn you during processing if it encounters information which comes from a Biblatex version which is incompatible. table 1 shows a compatibility matrix for the recent versions.

2 数据库指南

有必要区分一下 `bibtex` 程序和 `bibtex` 文件格式。`biblatex` 无论有无 `bibtex` 程序都可以使用，因为其默认使用的后端 `biber` 完全支持 `bibtex` 文件格式 (`bib`)。当使用 `biber` 后端时，用户仅需少量修改就可以使用 `bibtex` 数据文件。注意当使用 `bibtex` 后端时，无法使用任意的 `bst` 文件，因为该包基于一个特定的 `bibtex` 接口。当使用 `bibtex` 时，`biblatex` 只能使用自己的样式文件 (即通过 `biblatex` 宏包所定义的参考文献样式)。除非特殊说明，下面的条目指南适用于所有的后端。

本节描述在 `blx-dm.def` 中定义的默认数据模型。`blx-dm.def` 文件是宏包的一部分。数据模型的定义由 § 4.5.3 节中的宏实现。可以重新定义 `Biblatex` 和 `Biber` 所用的数据模型。因此数据源可以包括所有的条目类型和域 (当然这需要样式文件支持)。数据模型规范允许定义约束，因此数据源可以根据数据模型进行校验 (使用 `Biber` 的 `--validate_datamodel` 选项)。若需要定制数据模型，请参考 § 4.5.3 节。

2.1 条目类型

本节介绍默认数据类型支持的条目类型及每种条目类型支持的域。

2.1.1 常规类型

下面的列表说明了每种条目类型支持的域。注意，每种条目类型的预的使用是由参考文献样式决定的。下面的列表有两个目的，一是，说明有本包提供的标准样式支持的域，二是，作为定制样式的模板。注意，所谓需要的域并不是严格的必不可少，详见 § 2.3.2。而标记“可选”的域技术上是可选的。而参考文献样式规则常需要多于需要的域。默认的数据类型，定义了一些数据域的约束。比如 `ISBNs` 和一些特殊的域如 `gender`。但约束的校验只有在 `biber` 带校验选项时进行。通用域如 `abstract`, `annotation`, `label` 和 `shorthand` 并不在下面列表中，因为他们独立于条目类型，§ 2.2.3 节讨论的特殊域，同样，独立于条目类型，不在下面的列表中，默认的支持类型规范，见文件 `blx-dm.def`，内有 `Biblatex` 的完整规范。

article 指期刊，杂志，报纸，或其他周期性刊物的文章。刊物名在 `journaltitle` 域中，If the issue has its own title in addition to the main title of the periodical, it goes in the `issuetitle` field. 注意：刊物的 `editor` 及相关域，指的是期刊的，而 `translator` 及其相关域指的是文章的。

需要域: `author`, `title`, `journaltitle`, `year/date`。

可选域: `translator`, `annotator`, `commentator`, `subtitle`, `titleaddon`, `editor`, `editora`, `editorb`, `editorc`, `journalsubtitle`, `issuetitle`, `issuesubtitle`, `language`, `origlanguage`, `series`, `volume`, `number`, `eid`, `issue`, `month`, `pages`, `version`, `note`, `issn`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`。

book 单卷书有一个或多个作者。这些作者作为整体共享该工作，该条目涵盖了传统 `bibtex` 中的 `@inbook` 类型，详见 § 2.3.1。需要域: `author`, `title`, `year/date` 可选

域: editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

mvbook 多卷书，为了向后兼容，多卷书也可用@book类，然而建议最好使用专用类@mvbook。

需要域: author, title, year/date。可选域: editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, language, origlanguage, edition, volumes, series, number, note, publisher, location, isbn, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate。

inbook 书的一部分，具有自有单元和标题，注意：该类的定义不同于标准 BibTeX 给出的定义，见 § 2.3.1。

Required fields: author, title, booktitle, year/date

Optional fields: bookauthor, editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

bookinbook 类似于@inbook，但用于原本以单本书出版的工作，典型例子是：搜集一个作者的多个工作进行重印。

suppbook @book（书）的补充材料，与@inbook类密切相关，不同于@inbook 类是书的一部分，具有自己的标题，（比如，相同作者论文集中的一篇论文），这个类型常用于书的元素如：序，简介，前言，后记等，常有通用名。有时需要定制样式区别于@inbook。标准样式则认为它是@inbook的别名。

booklet 类似于书，但没有正式出版机构，若适当的话，使用howpublished域，可以自由格式提供出版信息。type域有时也很有用。

Required fields: author/editor, title, year/date

Optional fields: subtitle, titleaddon, language, howpublished, type, note, location, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

collection 单卷文集由多个具有不同标题和作者的独立工作构成，作为一个整体没有全局作者，常有一个编者。

Required fields: editor, title, year/date

Optional fields: editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

mvcollection 多卷文集，类似于 mvbook 之于 book

Required fields: editor, title, year/date

Optional fields: editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, language, origlanguage, edition, volumes, series, number, note, publisher, location, isbn, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

incollection 类似于 inbook 之于 book，author 指的是 title 的作者。editor 指的是 booktitle（即文集的标题）的编者。

Required fields: author, title, booktitle, year/date

Optional fields: editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

suppcollection 类似于 suppbok 之于 book。标准样式将其认为是@incollection 的别名。

manual 技术或其它文档，不必然是出版的形式，作者域 author 或编者域 editor 可忽略见 § 2.3.2。

Required fields: author/editor, title, year/date

Optional fields: subtitle, titleaddon, language, edition, type, series, number, version, note, organization, publisher, location, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

misc 备选类型，当条目无法分类为其他条目时，可使用。适当的话，使用 howpublished 域，可以自由格式提供出版信息。type 域有时也很有用。author, editor, 和 year 可忽略对于 § 2.3.2 节。

Required fields: author/editor, title, year/date

Optional fields: subtitle, titleaddon, language, howpublished, type, version, note, organization, location, date, month, year, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

online 在线资源, author (作者), editor (编者), 和 year (编者) 可忽略对于 § 2.3.2 节。该类用于网址等本身就是的在线资源, 注意: 所有类型都支持 url 域, 比如: 当增加一篇来自在线期刊的文章, 应优先使用 @article 类和它的 url 域。

Required fields: author/editor, title, year/date, url

Optional fields: subtitle, titleaddon, language, version, note, organization, date, month, year, addendum, pubstate, urldate

patent 专利或专利申请。号码或记录号在 number 域中给出, type 用于描述类型, location 用于描述专利范围, 如果存在暗指的范围不同的情况。注意, location 以 key list 的方式处理, 详见 § 2.2.1 节。

Required fields: author, title, number, year/date

Optional fields: holder, subtitle, titleaddon, type, version, location, note, date, month, year, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

periodical 周期性刊物的某一完整出版物, 比如某一期刊的特刊, 标题在 title 域中给出。如果该出版物有其自由标题, 除了在期刊的主标题外, 放在 issuetitle 域中。editor 域可忽略, 对于 § 2.3.2 节。

Required fields: editor, title, year/date

Optional fields: editora, editorb, editorc, subtitle, issuetitle, issuesubtitle, language, series, volume, number, issue, date, month, year, note, issn, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

supperperiodical 类似于 suppbok 之于 book, 在标准样式中与 @article 类相同处理, (可以认为 @article 类型就是 @inperiodical 类)。该类型应用于诸如: 常规栏目, 卜告, 或者给编者的信等具有通用名的资料很有用。

proceedings 单卷会议进展成果, 该类与 @collection 类似, 支持可选的 organization 域用于给出主办机构, editor 域可忽略对于 § 2.3.2 节。

Required fields: title, year/date

Optional fields: editor, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, eventtitle, eventtitleaddon, eventdate, venue, language, volume, part, volumes, series, number, note, organization, publisher, location, month, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

mvproceedings 多卷@proceedings条目，类似于 mvbook 之于 book。

Required fields: title, year/date

Optional fields: editor, subtitle, titleaddon, eventtitle, eventtitleaddon, eventdate, venue, language, volumes, series, number, note, organization, publisher, location, month, isbn, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

inproceedings 会议进展中的一篇文章，类似于 inbook 之于 book，支持可选organization域。

Required fields: author, title, booktitle, year/date

Optional fields: editor, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, eventtitle, eventtitleaddon, eventdate, venue, language, volume, part, volumes, series, number, note, organization, publisher, location, month, isbn, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

reference 单卷参考文献集，诸如百科全书或词典，它是通用@collection类的特殊变种，标准样式中与@collection相同处理。

mvreference 多卷文集献，标准样式中是@mvcollection的别名，类似于 mvbook 之于 book。

inreference 参考文献集中的一篇文章，类似于@incollection类条目，标准样式中与@incollection相同处理。

report 技术报告，研究报告或大学和一些机构的白皮书。用type域明确报告的类型，主办机构institution域中。

Required fields: author, title, type, institution, year/date

Optional fields: subtitle, titleaddon, language, number, version, note, location, month, isrn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

set 条目集，是一种特殊类型条目，详见 see § 3.11.5节。

thesis 学位论文用于在教育机构中满足学位要求，用type域规定论文的类型。

Required fields: author, title, type, institution, year/date

Optional fields: subtitle, titleaddon, language, note, location, month, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

unpublished 没有正式发表的有作者和标题的工作，如手稿，讲话稿等。需要的话，可使用howpublished域和note域，以自由格式提供附加信息。

Required fields: author, title, year/date

Optional fields: subtitle, titleaddon, language, howpublished, note, location, isbn, date, month, year, addendum, pubstate, url, urldate

xdata 特殊类型, @xdata 条目记录可能继承自使用xdata域的其他条目的数据, 这一类型只是作为数据容器, 不可被引用或加入到参考文献中, 详见 § 3.11.6节。 Biber only

custom[a-f] 用于特殊参考文献样式的自定义类型, 标准样式中不使用。

2.1.2 类型别名

本节中列出的类型用于向下兼容传统的类型。这些别名由后端在数据处理时一并处理, 样式中仅能见到这些别名所指代的类型, 而一种未知的类型一般输出为@misc条目。

conference BibTeX 遗留的@inproceedings的别名。

electronic @online的别名。

mastersthesis 类似于@thesis除了type域是可选的, 默认为'Master's thesis'。可使用type域重定义。

phdthesis 类似于@thesis除了type域是可选的, 默认为'PhD thesis'。可使用type域重定义。

techreport 类似于@report除了type域是可选的, 默认为'technical report'。可使用type域重定义。

www @online的别名, 用于兼容jurabib。

2.1.3 不支持的条目类型

本节中的条目类型类似于自定义类型, 标准样式不支持这些类型, 若使用标准样式, 他们将会以@misc条目类处理。

artwork 可视艺术作品, 如绘画, 雕像, 装饰。

audio 声音记录, 典型如: 有声 CD, dvd, 有声磁带或类似媒介, 也可参考@music。

bibnote 该类条目不象其它类型用于bib文件中, 它用于第三方宏包比如notes2bib。注赢放在note域中。要注意: @bibnote类型与命令\defbibnote没有任何关系。 \defbibnote命令用于在参考文献开头或结尾添加评注, 而@bibnote作为条目应用于提供尾注的宏包。

commentary 评注(集注), 与普通书籍的差异在于它具有法律效力, 如法律评注。

image 照片, 图片类媒介。

jurisdiction 法院记录, 如判决和类似文书。

legislation 法律, 提案, 立法建议或类似东西。

legal	如条约等法律文书。
letter	个人通信，如信，电邮，备忘等。
movie	电影。也可参见@video。
music	音乐记录，@audio的一种具体形式。
performance	音乐或戏剧表演和其它一些表演艺术工作。这一条目类型指的是表演的事件，而不是记录，打分或剧本。
review	工作回顾，是@article 的更具体形式。标准样式中以@article处理。
software	电脑软件。
standard	国内和国际标准，由国内和国际的标准组织（如国际标准组织）发布。
video	音视频，典型如：dvd，VHS 等媒介，可参见@movie。

2.2 条目域 Entry Fields

本节给出了 biblatex 默认数据模型支持域的概览，数据模型使用的数据类的介绍参考 § 2.2.1 小节，实际的域的列表见 §§ 2.2.2 and 2.2.3 小节。

2.2.1 数据类型 Data Types

在数据源文件如一个bib文件中，所有的参考文献数据都是以域的形式给出的。其中一些域，比如 author 和 editor 可能包含一些项的列表 (LIST)。这种列表结构由 BibTeX 文件格式通过关键词‘and’实现，该关键词用例分隔列表中的各个独立项。Biblatex 包实现了三种不同的数据类型用于处理参考文献数据：姓名列表 (name lists)，逐字列表 (literal lists)，和域 (fields)。同时也有几种列表的域的子类和一个内容类 (content)，可以用来从语义上区分那些无法从它们的数据类进行区分的域（见 § 4.5.3）。本节给出包所支持的数据类的概览。更多的关于 BibTeX 文件格式域到 Biblatex 数据类型的映射信息参考 §§ 2.2.2 and 2.2.3 小节。

Name lists 对姓名列表解析能以 and 为间隔符分成为独立的项。列表中的每一项进一步剖析为 4 个名称的组成部分，the first name（西方人名的第一个字，对于中文来说，名字无需划分，如果是拼音的话，是可以对应的划分的），the name prefix（姓名前缀比如 von, van, of, da, de, della, ...），the last name（姓），the name suffix（名字后缀，比如 junior, senior, ...）。姓名列表在 bib 文件中可能会用关键词‘and others’截断。典型的姓名列表是 author 和 editor。（译者注：在些 bib 文件中参考文献数据的时候，当有些机构的名称中含有空格的情况下，最好将整个机构用 {} 包含起来。）

利用 Biber，在默认的数据模型中姓名列表自动进行一个 \ifuse* 测试（见 § 4.6.2）。同时也自动创建了一个 ifuse* 选项用例控制姓名的标记和排序行为（见 § 3.1.3.1）。

Biber only

Literal lists are parsed and split up into the individual items at the `and` delimiter but not dissected further. Literal lists may be truncated in the `bib` file with the keyword `'and others'`. There are two subtypes:

逐字列表类似的解析并划分成独立的项，但不再进一步的剖析。逐字列表也类似的被截断。它有两个子类：

Literal lists in the strict sense are handled as described above. The individual items are simply printed as is. Typical examples of such literal lists are `publisher` and `location`.

严格的按照上面的描述处理。各独立项简单如实打印。典型的逐字列表是`publisher`和`location`。

Key lists are a variant of literal lists which may hold printable data or localization keys. For each item in the list, a test is performed to determine whether it is a known localization key (the localization keys defined by default are listed in § 4.9.2). If so, the localized string is printed. If not, the item is printed as is. A typical example of a key list is `language`.

关键词列表是逐字列表的变种，可以处理可打印的数据和本地化的关键词。对于列表中的每一项，首先执行测试它是否是已知的本地化关键词（本地化关键词默认定义在 § 4.9.2 中）。如果是，那么打印本地化的字符串，否则如实打印。典型的关键词列表是`language`。

Fields are usually printed as a whole. There are several subtypes:

域是以整体打印的，有多种子类：

Literal fields are printed as is. Typical examples of literal fields are `title` and `note`.

逐字域，如实打印。典型的逐字域是`title`和`note`。

Range fields consist of one or more ranges where all dashes are normalized and replaced by the command `\bibrangedash`. A range is something optionally followed by one or more dashes optionally followed by some non-dash (e.g. `5-7`). Any number of consecutive dashes will only yield a single range dash. A typical example of a range field is the `pages` field. See also the `\bibrangessep` command which can be used to customise the separator between multiple ranges. With Biber, range fields will be skipped and will generate a warning if they do not consist of one or more ranges. You can normalise messy range fields before they are parsed using `\DeclareSourcemap` (see § 4.5.2).

Biber only

范围域，包括一个或更多的范围，其中所有的短划线都规范化用命令`\bibrangedash`代替。一个范围是非短划线紧跟一个或多个短划线再跟上非短划线构成（比如 `5-7`）。任何有连续短划线的数字都将产生单个的范围短划线。典型的范围域是`pages`域。`\bibrangessep`命令可以自定义多个范围之间的分隔符。利用 `biber`，范围域将跳过

Biber only

并产生一个警告信息当它不是由一个或多个范围组成。可以用使用`\DeclareSourceMap`命令在其被解析之前对其进行规范化（见 § 4.5.2）。

Integer fields hold unformatted integers which may be converted to ordinals or strings as they are printed. A typical example is the `extrayear` field. 整数域，处理非格式化的整数，这些整数可以转化为序数或者字符串。典型的域有：`extrayear`。

Datepart fields hold unformatted integers which may be converted to ordinals or strings as they are printed. A typical example is the `month` field. 部分日期域，处理非格式化的整数，这些整数可以转化为序数或者字符串。典型的域有：`month`。

Date fields hold a date specification in `yyyy-mm-dd` format or a date range in `yyyy-mm-dd/yyyy-mm-dd` format. Date fields are special in that the date is parsed and split up into its components. See § 2.3.8 for details. A typical example is the `date` field.

日期域，处理一个以 `yyyy-mm-dd` 格式规定的日期或者一个以 `yyyy-mm-dd/yyyy-mm-dd` 格式规定的日期范围。日期域需要特别的解析并分成不同的部分。更多细节见 § 2.3.8。典型的域有：`date`。

Verbatim fields are processed in verbatim mode and may contain special characters. Typical examples of verbatim fields are `file` and `doi`. 字面（逐字）域，以 verbatim 模式处理可能包含特殊的字符，典型的字面（逐字）域有：`file`和`doi`。

URI fields are processed in verbatim mode and may contain special characters. They are also URL-escaped if they don't look like they already are. The typical example of a uri field is `url`.

`url` 域以 verbatim 模式处理，可能包含特殊的字符。如果它们看起来不像其实质，也可以例外处理。典型域是 `url`。

Separated value fields A separated list of literal values. Examples are the `keywords` and `options` fields. The separator is always a comma when using BibTeX as the backend but can be configured to be any Perl regular expression when using Biber via the `xsvsep` option which defaults to the usual BibTeX comma surrounded by optional whitespace.

分离值域，字母值的分离列表。典型的域是`keywords`和`options`。使用 BibTeX 后端时时分隔符总是逗号，但可以使用`xsvsep`选项重设为任何的 PERL 正则表达式，当使用 biber 后端时，当然默认情况下也是逗号或者逗号加空格。

Pattern fields A literal field which must match a particular pattern. An example is the `gender` field from § 2.2.3.

型（pattern）域，是逐字域必须匹配特殊的 pattern。典型的域是 `gender`，见 § 2.2.3。

Key fields May hold printable data or localization keys. A test is performed to determine whether the value of the field is a known localization key (the localization keys defined by default are listed in § 4.9.2). If so, the localized string is printed. If not, the value is printed as is. A typical example is the `type` field.

关键词域，可以处理可打印数据或本地化关键词。类似于逐字列表中的关键词列表，这是这里是单个的域，而不是列表。

Code fields Holds TeX code.

代码（code）域，处理代码。

2.2.2 数据域 Data Fields

The fields listed in this section are the regular ones holding printable data in the default data model. The name on the left is the default data model name of the field as used by Biblatex and its backend. The Biblatex data type is given to the right of the name. See § 2.2.1 for explanation of the various data types.

本节中列出的域是默认数据模型携带可打印数据的。左侧给出的名称是 Biblatex 及其后端中使用的域的默认数据模型名。而右侧给出的是它的数据类型。不同的数据类型的解释见 § 2.2.1。

Some fields are marked as ‘Label fields’ which means that they are often used as abbreviation labels when printing bibliography lists in the sense of section § 3.6.4. Biblatex automatically creates supporting macros for such fields. See § 3.6.4.

Biber only

一些以 ‘Label fields’ 标记的域表示当以 § 3.6.4 节的方式打印参考文献列表时它们常用为简化标签。Biblatex 自动创建支持这些域的宏，详见 § 3.6.4。

Biber only

abstract field (literal)

This field is intended for recording abstracts in a `bib` file, to be printed by a special bibliography style. It is not used by all standard bibliography styles.

标准样式不使用

addendum field (literal)

Miscellaneous bibliographic data to be printed at the end of the entry. This is similar to the `note` field except that it is printed at the end of the bibliography entry.

afterword list (name)

The author(s) of an afterword to the work. If the author of the afterword is identical to the editor and/or translator, the standard styles will automatically concatenate these fields in the bibliography. See also `introduction` and `foreword`.

后记的作者，当域编者或译者相同时，标准样式自动联接这些域。

annotation field (literal)

This field may be useful when implementing a style for annotated bibliographies. It is not used by all standard bibliography styles. Note that this field is completely unrelated to `annotator`. The `annotator` is the author of annotations which are part of the work cited.

标准样式不使用

annotator list (name)

The author(s) of annotations to the work. If the `annotator` is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `commentator`.

注释的作者，当与编者和译者同时，标准样式自动联接。

author list (name)

The author(s) of the `title`.

作者，姓名类

authortype field (key)

The type of author. This field will affect the string (if any) used to introduce the author. Not used by the standard bibliography styles.

标准样式不用

bookauthor list (name)

The author(s) of the `booktitle`.

书名作者，姓名类

bookpagination field (key)

If the work is published as part of another one, this is the pagination scheme of the enclosing work, i.e., `bookpagination` relates to pagination like `booktitle` to `title`. The value of this field will affect the formatting of the `pages` and `pagetotal` fields. The key should be given in the singular form. Possible keys are `page`, `column`, `line`, `verse`, `section`, and `paragraph`. See also `pagination` as well as § 2.3.10.

标记页数

booksubtitle field (literal)

The subtitle related to the `booktitle`. If the `subtitle` field refers to a work which is part of a larger publication, a possible subtitle of the main work is given in this field. See also `subtitle`.

工作的子标题

booktitle field (literal)

If the `title` field indicates the title of a work which is part of a larger publication, the title of the main work is given in this field. See also `title`.

书的标题，当 `title` 用来表示该书所在的更大的出版物的标题时用

booktitleaddon field (literal)

An annex to the `booktitle`, to be printed in a different font.

附加标题

chapter field (literal)

A chapter or section or any other unit of a work.

章节名

commentator list (name)

The author(s) of a commentary to the work. Note that this field is intended for commented editions which have a commentator in addition to the author. If the work is a stand-alone commentary, the commentator should be given in the `author` field. If the commentator is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `annotator`.

某工作的评论的作者，类似于 `annotator`。

date field (date)

The publication date. See also `month` and `year` as well as § 2.3.8.

出版日期。还可参见 `month`, `year` 和 § 2.3.8 小节。

doi field (verbatim)

The Digital Object Identifier of the work.

数字对象标识符。verbatim 类。

edition field (integer or literal)

The edition of a printed publication. This must be an integer, not an ordinal. Don't say `edition={First}` or `edition={1st}` but `edition={1}`. The bibliography style converts this to a language dependent ordinal. It is also possible to give the edition as a literal string, for example "Third, revised and expanded edition".

版本域，整数或者逐字类。

editor list (name)

The editor(s) of the `title`, `booktitle`, or `maintitle`, depending on the entry type. Use the `editortype` field to specify the role if it is different from 'editor'. See § 2.3.6 for further hints.

编者，姓名类

editora list (name)

A secondary editor performing a different editorial role, such as compiling, redacting, etc. Use the `editoratype` field to specify the role. See § 2.3.6 for further hints.

editora, 次要编者, 比如汇集, 编纂, 修订等工作作者。

editorb list (name)

Another secondary editor performing a different role. Use the `editorbtype` field to specify the role. See § 2.3.6 for further hints.

editora, 次要编者。

editorc list (name)

Another secondary editor performing a different role. Use the `editorctype` field to specify the role. See § 2.3.6 for further hints.

editora, 次要编者。

editortype field (key)

The type of editorial role performed by the `editor`. Roles supported by default are `editor`, `compiler`, `founder`, `continuator`, `redactor`, `reviser`, `collaborator`. The role ‘`editor`’ is the default. In this case, the field is omissible. See § 2.3.6 for further hints.

编者类型, 用于描述编者的角色。

editoratype field (key)

Similar to `editortype` but referring to the `editora` field. See § 2.3.6 for further hints.

editoratype, 次要编者类型, 用于描述编者的角色。

editorbtype field (key)

Similar to `editortype` but referring to the `editorb` field. See § 2.3.6 for further hints.

editorbtype, 次要编者类型, 用于描述编者的角色。

editorctype field (key)

Similar to `editortype` but referring to the `editorc` field. See § 2.3.6 for further hints.

editorctype, 次要编者类型, 用于描述编者的角色。

eid field (literal)

The electronic identifier of an `@article`.

文章电子 ID 号。

`entrysubtype` field (literal)

This field, which is not used by the standard styles, may be used to specify a subtype of an entry type. This may be useful for bibliography styles which support a finer-grained set of entry types.

标准样式不用。

`eprint` field (verbatim)

The electronic identifier of an online publication. This is roughly comparable to a DOI but specific to a certain archive, repository, service, or system. See § 3.11.7 for details. Also see `eprinttype` and `eprintclass`.

在线出版物的电子 ID 号。

`eprintclass` field (literal)

Additional information related to the resource indicated by the `eprinttype` field. This could be a section of an archive, a path indicating a service, a classification of some sort, etc. See § 3.11.7 for details. Also see `eprint` and `eprinttype`.

`eprinttype` 域的附加信息。

`eprinttype` field (literal)

The type of `eprint` identifier, e.g., the name of the archive, repository, service, or system the `eprint` field refers to. See § 3.11.7 for details. Also see `eprint` and `eprintclass`.

`eprint` 域的类型信息。比如档案，仓库，服务，系统的名字。

`eventdate` field (date)

The date of a conference, a symposium, or some other event in `@proceedings` and `@inproceedings` entries. This field may also be useful for the custom types listed in § 2.1.3. See also `eventtitle` and `venue` as well as § 2.3.8.

一些如会议，专题讨论，座谈等事件的日期。

`eventtitle` field (literal)

The title of a conference, a symposium, or some other event in `@proceedings` and `@inproceedings` entries. This field may also be useful for the custom types listed in § 2.1.3. Note that this field holds the plain title of the event. Things like “Proceedings of the Fifth XYZ Conference” go into the `titleaddon` or `booktitleaddon` field, respectively. See also `eventdate` and `venue`.

`eventtitleaddon` field (literal)

An annex to the `eventtitle` field. Can be used for known event acronyms, for example.

`file` field (verbatim)

A local link to a PDF or other version of the work. Not used by the standard bibliography styles.

标准样式不使用

`foreword` list (name)

The author(s) of a foreword to the work. If the author of the foreword is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `introduction` and `afterword`.

前言作者。

`holder` list (name)

The holder(s) of a `@patent`, if different from the `author`. Note that corporate holders need to be wrapped in an additional set of braces, see § 2.3.3 for details. This list may also be useful for the custom types listed in § 2.1.3.

专利持有人。

`howpublished` field (literal)

A publication notice for unusual publications which do not fit into any of the common categories.

非常规出版物的出版信息。

`indextitle` field (literal)

A title to use for indexing instead of the regular `title` field. This field may be useful if you have an entry with a title like “An Introduction to ...” and want that indexed as “Introduction to ..., An”. Style authors should note that BibLaTeX automatically copies the value of the `title` field to `indextitle` if the latter field is undefined.

`institution` list (literal)

The name of a university or some other institution, depending on the entry type. Traditional BibTeX uses the field name `school` for theses, which is supported as an alias. See also §§ 2.2.5 and 2.3.4.

大学或研究机构，`bibtex` 中使用的域名是 `school`。

`introduction` list (name)

The author(s) of an introduction to the work. If the author of the introduction is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `foreword` and `afterword`.

简介的作者。

isan field (literal)

The International Standard Audiovisual Number of an audiovisual work. Not used by the standard bibliography styles.

ISAN 号。

isbn field (literal)

The International Standard Book Number of a book.

ISBN 号。

ismn field (literal)

The International Standard Music Number for printed music such as musical scores. Not used by the standard bibliography styles.

ISMN 号。

isrn field (literal)

The International Standard Technical Report Number of a technical report.

ISRN 号。

issn field (literal)

The International Standard Serial Number of a periodical.

ISSN 号。

issue field (literal)

The issue of a journal. This field is intended for journals whose individual issues are identified by a designation such as ‘Spring’ or ‘Summer’ rather than the month or a number. Since the placement of `issue` is similar to `month` and `number`, this field may also be useful with double issues and other special cases. See also `month`, `number`, and § 2.3.9.

期刊的版次，通常是特殊的版次，而不是以月份和数字标示的版次。（译者注：可能类似于什么什么增刊，特刊等。）

issuesubtitle field (literal)

The subtitle of a specific issue of a journal or other periodical.

issuetitle field (literal)

The title of a specific issue of a journal or other periodical.

iswc field (literal)

The International Standard Work Code of a musical work. Not used by the standard bibliography styles.

ISWC 码。

`journalsubtitle` field (literal)

The subtitle of a journal, a newspaper, or some other periodical.

期刊等周期出版物的子名

`journaltitle` field (literal)

The name of a journal, a newspaper, or some other periodical.

期刊等周期出版物的名

`label` field (literal)

A designation to be used by the citation style as a substitute for the regular label if any data required to generate the regular label is missing. For example, when an author-year citation style is generating a citation for an entry which is missing the author or the year, it may fall back to `label`. See § 2.3.2 for details. Note that, in contrast to `shorthand`, `label` is only used as a fallback. See also `shorthand`.

`language` list (key)

The language(s) of the work. Languages may be specified literally or as localization keys. If localization keys are used, the prefix `lang` is omissible. See also `origlanguage` and compare `langid` in § 2.2.3.

工作的语言。

`library` field (literal)

This field may be useful to record information such as a library name and a call number. This may be printed by a special bibliography style if desired. Not used by the standard bibliography styles.

标准样式不用。

`location` list (literal)

The place(s) of publication, i.e., the location of the publisher or institution, depending on the entry type. Traditional BibTeX uses the field name `address`, which is supported as an alias. See also §§ 2.2.5 and 2.3.4. With `@patent` entries, this list indicates the scope of a patent. This list may also be useful for the custom types listed in § 2.1.3.

出版社地址。

`mainsubtitle` field (literal)

The subtitle related to the `maintitle`. See also `subtitle`.

maintitle field (literal)

The main title of a multi-volume book, such as *Collected Works*. If the `title` or `booktitle` field indicates the title of a single volume which is part of multi-volume book, the title of the complete work is given in this field.

集合物的名称，当 `title` 指单卷物的时候用。

maintitleaddon field (literal)

An annex to the `maintitle`, to be printed in a different font.

month field (datepart)

The publication month. This must be an integer, not an ordinal or a string. Don't say `month={January}` but `month={1}`. The bibliography style converts this to a language dependent string or ordinal where required. See also `date` as well as §§ 2.3.9 and 2.3.8.

出版月份。

nameaddon field (literal)

An addon to be printed immediately after the author name in the bibliography. Not used by the standard bibliography styles. This field may be useful to add an alias or pen name (or give the real name if the pseudonym is commonly used to refer to that author).

标准样式不用。

note field (literal)

Miscellaneous bibliographic data which does not fit into any other field. The `note` field may be used to record bibliographic data in a free format. Publication facts such as “Reprint of the edition London 1831” are typical candidates for the `note` field. See also `addendum`.

混杂数据，无法分类到某一种域中。

number field (literal)

The number of a journal or the volume/number of a book in a `series`. See also `issue` as well as §§ 2.3.7 and 2.3.9. With `@patent` entries, this is the number or record token of a patent or patent request.

期刊或数的卷的数字号比如期号。在专利条目中是专利申请号。

organization list (literal)

The organization(s) that published a `@manual` or an `@online` resource, or sponsored a conference. See also § 2.3.4.

手册或在线资源或会议主办的机构。

`origdate` field (date)

If the work is a translation, a reprint, or something similar, the publication date of the original edition. Not used by the standard bibliography styles. See also `date`.

标准样式不用。

`origlanguage` field (key)

If the work is a translation, the language of the original work. See also `language`.

`origlocation` list (literal)

If the work is a translation, a reprint, or something similar, the location of the original edition. Not used by the standard bibliography styles. See also `location` and § 2.3.4.

`origpublisher` list (literal)

If the work is a translation, a reprint, or something similar, the publisher of the original edition. Not used by the standard bibliography styles. See also `publisher` and § 2.3.4.

`origtitle` field (literal)

If the work is a translation, the title of the original work. Not used by the standard bibliography styles. See also `title`.

`pages` field (range)

One or more page numbers or page ranges. If the work is published as part of another one, such as an article in a journal or a collection, this field holds the relevant page range in that other work. It may also be used to limit the reference to a specific part of a work (a chapter in a book, for example).

页码范围。

`pagetotal` field (literal)

The total number of pages of the work.

总页数。

`pagination` field (key)

The pagination of the work. The value of this field will affect the formatting the `<postnote>` argument to a citation command. The key should be given in the singular form. Possible keys are `page`, `column`, `line`, `verse`, `section`, and `paragraph`. See also `bookpagination` as well as §§ 2.3.10 and 3.12.3.

part field (literal)

The number of a partial volume. This field applies to books only, not to journals. It may be used when a logical volume consists of two or more physical ones. In this case the number of the logical volume goes in the `volume` field and the number of the part of that volume in the `part` field. See also `volume`.

卷的部分的号。

publisher list (literal)

The name(s) of the publisher(s). See also § 2.3.4.

出版社名。

pubstate field (key)

The publication state of the work, e. g., ‘in press’. See § 4.9.2.11 for known publication states.

出版状态。

reprinttitle field (literal)

The title of a reprint of the work. Not used by the standard styles.

BibTeX only

标准样式不用。

series field (literal)

The name of a publication series, such as “Studies in ...”, or the number of a journal series. Books in a publication series are usually numbered. The number or volume of a book in a series is given in the `number` field. Note that the `@article` entry type makes use of the `series` field as well, but handles it in a special way. See § 2.3.7 for details.

系列丛书的标题。

shortauthor list (name)

Label field

The author(s) of the work, given in an abbreviated form. This field is mainly intended for abbreviated forms of corporate authors, see § 2.3.3 for details.

缩略的作者名。

shorteditor list (name)

Label field

The editor(s) of the work, given in an abbreviated form. This field is mainly intended for abbreviated forms of corporate editors, see § 2.3.3 for details.

缩略的编者名。

shorthand field (literal)

Label field

A special designation to be used by the citation style instead of the usual label. If defined, it overrides the default label. See also `label`.

特殊标记用于引用样式代替常用的标签。

shorthandintro field (literal)

The verbose citation styles which comes with this package use a phrase like “henceforth cited as [shorthand]” to introduce shorthands on the first citation. If the `shorthandintro` field is defined, it overrides the standard phrase. Note that the alternative phrase must include the shorthand.

shortjournal field (literal)

Label field

A short version or an acronym of the `journaltitle`. Not used by the standard bibliography styles.

期刊名首字母缩略词。

shortseries field (literal)

Label field

A short version or an acronym of the `series` field. Not used by the standard bibliography styles.

shorttitle field (literal)

Label field

The title in an abridged form. This field is usually not included in the bibliography. It is intended for citations in author-title format. If present, the author-title citation styles use this field instead of `title`.

subtitle field (literal)

The subtitle of the work.

工作的子标题。

title field (literal)

The title of the work.

工作的标题。

titleaddon field (literal)

An annex to the `title`, to be printed in a different font.

translator list (name)

The translator(s) of the `title` or `booktitle`, depending on the entry type. If the translator is identical to the `editor`, the standard styles will automatically concatenate these fields in the bibliography.

译者。

type field (key)

The type of a manual, patent, report, or thesis. This field may also be useful for the custom types listed in § 2.1.3.

类型。

url field (url)

The URL of an online publication. If it is not URL-escaped (no ‘%’ chars), with Biber, it will be URI-escaped according to RFC 3987, that is, even Unicode chars will be correctly escaped.

url。

urldate field (date)

The access date of the address specified in the `url` field. See also § 2.3.8.

venue field (literal)

The location of a conference, a symposium, or some other event in `@proceedings` and `@inproceedings` entries. This field may also be useful for the custom types listed in § 2.1.3. Note that the `location` list holds the place of publication. It therefore corresponds to the `publisher` and `institution` lists. The location of the event is given in the `venue` field. See also `eventdate` and `eventtitle`.

现场。会议，座谈，或其它事件的地址。

version field (literal)

The revision number of a piece of software, a manual, etc.

版本。

volume field (literal)

The volume of a multi-volume book or a periodical. See also `part`.

期刊或多卷书的卷的卷数。

volumes field (literal)

The total number of volumes of a multi-volume work. Depending on the entry type, this field refers to `title` or `maintitle`.

总卷数。

year field (literal)

The year of publication. See also `date` and § 2.3.8.

年份。

2.2.3 特殊域 Special Fields

The fields listed in this section do not hold printable data but serve a different purpose. They apply to all entry types in the default data model.

本节中的域不包括可打印的数据，用于其它用途，可用于默认数据模型的所有条目类型。

`crossref` field (entry key)

This field holds an entry key for the cross-referencing feature. Child entries with a `crossref` field inherit data from the parent entry specified in the `crossref` field. If the number of child entries referencing a specific parent entry hits a certain threshold, the parent entry is automatically added to the bibliography even if it has not been cited explicitly. The threshold is settable with the `mincrossrefs` package option from § 3.1.2.1. Style authors should note that whether or not the `crossref` fields of the child entries are defined on the Biblatex level depends on the availability of the parent entry. If the parent entry is available, the `crossref` fields of the child entries will be defined. If not, the child entries still inherit the data from the parent entry but their `crossref` fields will be undefined. Whether the parent entry is added to the bibliography implicitly because of the threshold or explicitly because it has been cited does not matter. See also the `xref` field in this section as well as § 2.4.1.

`entryset` field (separated values)

This field is specific to entry sets. See § 3.11.5 for details. This field is consumed by the backend processing and does not appear in the `.bbl`.

`execute` field (code)

A special field which holds arbitrary TeX code to be executed whenever the data of the respective entry is accessed. This may be useful to handle special cases. Conceptually, this field is comparable to the hooks `\AtEveryBibitem`, `\AtEveryLositem`, and `\AtEveryCitekey` from § 4.10.6, except that it is definable on a per-entry basis in the `bib` file. Any code in this field is executed automatically immediately after these hooks.

`gender` field (Pattern matching one of: `sf`, `sm`, `sn`, `pf`, `pm`, `pn`, `pp`)

The gender of the author or the gender of the editor, if there is no author. The following identifiers are supported: `sf` (feminine singular, a single female name), `sm` (masculine singular, a single male name), `sn` (neuter singular, a single neuter name), `pf` (feminine plural, a list of female names), `pm` (masculine plural, a list of male names), `pn` (neuter plural, a list of neuter names), `pp` (plural, a mixed gender list of names). This information is only required by special bibliography and citation styles and only in certain languages. For example, a citation style may replace recurrent author names with a term such as ‘idem’. If the Latin word is used, as is custom in English and French, there

Language	Region/Dialect	Identifiers
Catalan	Spain, France, Andorra, Italy	catalan
Croatian	Croatia, Bosnia and Herzegovina, Serbia	croatian
Czech	Czech Republic	czech
Danish	Denmark	danish
Dutch	Netherlands	dutch
English	USA	american, USenglish, english
	United Kingdom	british, UKenglish
	Canada	canadian
	Australia	australian
	New Zealand	newzealand
Finnish	Finland	finnish
French	France, Canada	french
German	Germany	german
	Austria	austrian
German (new)	Germany	ngerman
	Austria	naustrian
Greek	Greece	greek
Italian	Italy	italian
Norwegian	Norway	norwegian, norsk, nynorsk
Polish	Poland	polish
Portuguese	Brazil	brazil
	Portugal	portuguese, portuges
Russian	Russia	russian
Slovene	Slovenian	slovene
Spanish	Spain	spanish
Swedish	Sweden	swedish

Table 2: Supported Languages

is no need to specify the gender. In German publications, however, such key terms are usually given in German and in this case they are gender-sensitive.

langid field (identifier)

The language id of the bibliography entry. The alias `hyphenation` is provided for backwards compatibility. The identifier must be a language name known to the `babel/polyglossia` packages. This information may be used to switch hyphenation patterns and localize strings in the bibliography. Note that the language names are case sensitive. The languages currently supported by this package are given in table 2. Note that `babel` treats the identifier `english` as an alias for `british` or `american`, depending on the `babel` version. The `Biblatex` package always treats it as an alias for `american`. It is preferable to use the language identifiers `american` and `british` (`babel`) or a language specific option to specify a language variant (`polyglossia`, using the `langidopts` field) to avoid any possible confusion. Compare language in § 2.2.2.

langidopts field (literal)

For `polyglossia` users, allows per-entry language specific options. The literal value of this field is passed to `polyglossia`'s language switching facility when using the package option `autolang=langname`. For example, the fields:

```
langid          = {english},  
langidopts      = {variant=british},
```

would wrap the bibliography entry in:

```
\english[variant=british]  
...  
\endenglish
```

ids field (separated list of entrykeys)

Biber only

Citation key aliases for the main citation key. An entry may be cited by any of its aliases and `Biblatex` will treat the citation as if it had used the primary citation key. This is to aid users who change their citation keys but have legacy documents which use older keys for the same entry. This field is consumed by the backend processing and does not appear in the `.bbl`.

indexsorttitle field (literal)

The title used when sorting the index. In contrast to `indextitle`, this field is used for sorting only. The printed title in the index is the `indextitle` or the `title` field. This field may be useful if the title contains special characters or commands which interfere with the sorting of the index. Consider this example:

```

title          = {The \LaTeX\ Companion},
indextitle     = {\LaTeX\ Companion, The},
indexsorttitle = {LATEX Companion},

```

Style authors should note that Biblatex automatically copies the value of either the `indextitle` or the `title` field to `indexsorttitle` if the latter field is undefined.

keywords field (separated values)

A separated list of keywords. These keywords are intended for the bibliography filters (see §§ 3.6.2 and 3.11.4), they are usually not printed. Note that with the default separator (comma), spaces around the separator are ignored.

options field (separated $\langle key \rangle = \langle value \rangle$ options)

A separated list of entry options in $\langle key \rangle = \langle value \rangle$ notation. This field is used to set options on a per-entry basis. See § 3.1.3 for details. Note that citation and bibliography styles may define additional entry options.

presort field (string)

A special field used to modify the sorting order of the bibliography. This field is the first item the sorting routine considers when sorting the bibliography, hence it may be used to arrange the entries in groups. This may be useful when creating subdivided bibliographies with the bibliography filters. Please refer to § 3.5 for further details. Also see § 4.5.5. This field is consumed by the backend processing and does not appear in the `.bbl`.

related field (separated values)

Biber only

Citation keys of other entries which have a relationship to this entry. The relationship is specified by the `relatedtype` field. Please refer to § 3.4 for further details.

relatedoptions field (separated values)

Biber only

Per-type options to set for a related entry. Note that this does not set the options on the related entry itself, only the `dataonly` clone which is used as a datasource for the parent entry.

relatedtype field (identifier)

Biber only

An identifier which specified the type of relationship for the keys listed in the `related` field. The identifier is a localized bibliography string printed before the data from the related entry list. It is also used to identify type-specific formatting directives and bibliography macros for the related entries. Please refer to § 3.4 for further details.

relatedstring field (literal)

Biber only

A field used to override the bibliography string specified by `relatedtype`. Please refer to § 3.4 for further details.

`sortkey` field (literal)

A field used to modify the sorting order of the bibliography. Think of this field as the master sort key. If present, Biblatex uses this field during sorting and ignores everything else, except for the `presort` field. Please refer to § 3.5 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

`sortname` list (name)

A name or a list of names used to modify the sorting order of the bibliography. If present, this list is used instead of `author` or `editor` when sorting the bibliography. Please refer to § 3.5 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

`sortshorthand` field (literal)

Biber only

Similar to `sortkey` but used in the list of shorthands. If present, Biblatex uses this field instead of `shorthand` when sorting the list of shorthands. This is useful if the `shorthand` field holds shorthands with formatting commands such as `\emph` or `\textbf`. This field is consumed by the backend processing and does not appear in the `.bbl`.

`sorttitle` field (literal)

A field used to modify the sorting order of the bibliography. If present, this field is used instead of the `title` field when sorting the bibliography. The `sorttitle` field may come in handy if you have an entry with a title like “An Introduction to...” and want that alphabetized under ‘T’ rather than ‘A’. In this case, you could put “Introduction to...” in the `sorttitle` field. Please refer to § 3.5 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

`sortyear` field (literal)

A field used to modify the sorting order of the bibliography. If present, this field is used instead of the `year` field when sorting the bibliography. Please refer to § 3.5 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

`xdata` field (separated list of entrykeys)

Biber only

This field inherits data from one or more `@xdata` entries. Conceptually, the `xdata` field is related to `crossref` and `xref`: `crossref` establishes a logical parent/child relation and inherits data; `xref` establishes as logical parent/child relation without inheriting data; `xdata` inherits data without establishing a relation. The value of the `xdata` may be a single entry key or a separated list of keys. See § 3.11.6 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

`xref` field (entry key)

This field is an alternative cross-referencing mechanism. It differs from `crossref` in that the child entry will not inherit any data from the parent entry specified in the `xref` field. If the number of child entries referencing a specific parent entry hits a certain threshold, the parent entry is automatically added to the bibliography even if it has not been cited explicitly. The threshold is settable with the `mincrossrefs` package option from § 3.1.2.1. Style authors should note that whether or not the `xref` fields of the child entries are defined on the Biblatex level depends on the availability of the parent entry. If the parent entry is available, the `xref` fields of the child entries will be defined. If not, their `xref` fields will be undefined. Whether the parent entry is added to the bibliography implicitly because of the threshold or explicitly because it has been cited does not matter. See also the `crossref` field in this section as well as § 2.4.1.

2.2.4 自定义域 Custom Fields

The fields listed in this section are intended for special bibliography styles. They are not used by the standard bibliography styles.

本节中的域用于特定的参考文献样式，标准样式不使用。

`name[a-c]` list (name)

Custom lists for special bibliography styles. Not used by the standard bibliography styles.

`name[a-c]type` field (key)

Similar to `authortype` and `editortype` but referring to the fields `name[a-c]`. Not used by the standard bibliography styles.

`list[a-f]` list (literal)

Custom lists for special bibliography styles. Not used by the standard bibliography styles.

`user[a-f]` field (literal)

Custom fields for special bibliography styles. Not used by the standard bibliography styles.

`verb[a-c]` field (literal)

Similar to the custom fields above except that these are verbatim fields. Not used by the standard bibliography styles.

2.2.5 域的别名 Field Aliases

The aliases listed in this section are provided for backwards compatibility with traditional BibTeX and other applications based on traditional BibTeX styles. Note that these aliases are immediately resolved as the `bib` file is processed. All bibliography and citation styles must use the names of the fields they point to, not the alias. In `bib` files, you may use either the alias or the field name but not both at the same time.

本级列出的别名用于兼容传统的 `bibtex`。在 `biblatex` 的样式文件中则必须使用这些别名所指代的域名。

`address` list (literal)

An alias for `location`, provided for BibTeX compatibility. Traditional BibTeX uses the slightly misleading field name `address` for the place of publication, i. e., the location of the publisher, while Biblatex uses the generic field name `location`. See §§ 2.2.2 and 2.3.4.

`address` 就是 `location`。

`annotate` field (literal)

An alias for `annotation`, provided for `jurabib` compatibility. See § 2.2.2.

`archiveprefix` field (literal)

An alias for `eprinttype`, provided for arXiv compatibility. See §§ 2.2.2 and 3.11.7.

`journal` field (literal)

An alias for `journaltitle`, provided for BibTeX compatibility. See § 2.2.2.

`journal` 就是 `journaltitle`。

`key` field (literal)

An alias for `sortkey`, provided for BibTeX compatibility. See § 2.2.3.

`key` 是 `sortkey`。

`pdf` field (verbatim)

An alias for `file`, provided for JabRef compatibility. See § 2.2.2.

`pdf` 是 `file`。

`primaryclass` field (literal)

An alias for `eprintclass`, provided for arXiv compatibility. See §§ 2.2.2 and 3.11.7.

`school` list (literal)

An alias for `institution`, provided for BibTeX compatibility. The `institution` field is used by traditional BibTeX for technical reports whereas the `school` field holds

the institution associated with theses. The Biblatex package employs the generic field name `institution` in both cases. See §§ 2.2.2 and 2.3.4.

`school` 是 `institution`。

2.3 使用注意事项 Usage Notes

The entry types and fields supported by this package should for the most part be intuitive to use for anyone familiar with BibTeX. However, apart from the additional types and fields provided by this package, some of the familiar ones are handled in a way which is in need of explanation.

本包支持的类型和域对于熟悉 `bibtex` 的人来说可以直觉的使用。然而除了传统的类型和域，其中某些熟悉的域和类型需要进一步解释。

This package includes some compatibility code for `bib` files which were generated with a traditional BibTeX style in mind. Unfortunately, it is not possible to handle all legacy files automatically because Biblatex's data model is slightly different from traditional BibTeX. Therefore, such `bib` files will most likely require editing in order to work properly with this package. In sum, the following items are different from traditional BibTeX styles:

本包包含了一些兼容 `bibtex` 的代码，但不可能完全处理所有的继承 `bib` 文件，因此这些 `bib` 需要修改，一些不同的需要修改的地方，总结如下：

- The entry type `@inbook`. See §§ 2.1.1 and 2.3.1 for details.
- The fields `institution`, `organization`, and `publisher` as well as the aliases `address` and `school`. See §§ 2.2.2, 2.2.5, 2.3.4 for details.
- The handling of certain types of titles. See § 2.3.5 for details.
- The field `series`. See §§ 2.2.2 and 2.3.7 for details.
- The fields `year` and `month`. See §§ 2.2.2, 2.3.8, 2.3.9 for details.
- The field `edition`. See § 2.2.2 for details.
- The field `key`. See § 2.3.2 for details.

Users of the `jurabib` package should note that the `shortauthor` field is treated as a name list by Biblatex, see § 2.3.3 for details.

2.3.1 `@inbook` 条目类型 The Entry Type `@inbook`

`@inbook` 条目类型只用来表示以书籍 (专著) 中包含的某一具有标题的部分作为引文的参考文献。它与 `@book` 的关系如同 `@incollection` 和 `@collection` 的关系。具体示例见 § 2.3.5。如果只是将书籍的章节作为引文，简单的使用 `book` 类型再加上 `chapter` 和/or `pages` 域即可。至于参考文献列表是否应该完全包含章节引文的问题是有争议的，因为章并不是一个参考文献实体。

2.3.2 缺失和可忽略数据 Missing and Omissible Data

The fields marked as ‘required’ in § 2.1.1 are not strictly required in all cases. The bibliography styles which ship with this package can get by with as little as a `title` field for most entry types. A book published anonymously, a periodical without an explicit editor, or a software manual without an explicit author should pose no problem as far as the bibliography is concerned. Citation styles, however, may have different requirements. For example, an author-year citation scheme obviously requires an `author/``editor` and a `year` field.

You may generally use the `label` field to provide a substitute for any missing data required for citations. How the `label` field is employed depends on the citation style. The author-year citation styles which come with this package use the `label` field as a fallback if either the `author/editor` or the `year` is missing. The numeric styles, on the other hand, do not use it at all since the numeric scheme is independent of the available data. The author-title styles ignore it as well, because the bare `title` is usually sufficient to form a unique citation and a title is expected to be available in any case. The `label` field may also be used to override the non-numeric portion of the automatically generated `labelalpha` field used by alphabetic citation styles. See § 4.2.4 for details.

Note that traditional BibTeX styles support a `key` field which is used for alphabetizing if both `author` and `editor` are missing. The Biblatex package treats `key` as an alias for `sortkey`. In addition to that, it offers very fine-grained sorting controls, see §§ 2.2.3 and 3.5 for details. The `natbib` package employs the `key` field as a fallback label for citations. Use the `label` field instead.

2.3.3 集体作者和编者 Corporate Authors and Editors

Corporate authors and editors are given in the `author` or `editor` field, respectively. Note that they must be wrapped in an extra pair of curly braces to prevent data parsing from treating them as personal names which are to be dissected into their components. Use the `shortauthor` field if you want to give an abbreviated form of the name or an acronym for use in citations.

```
author      = {{National Aeronautics and Space Administration}},
shortauthor = {NASA},
```

The default citation styles will use the short name in all citations while the full name is printed in the bibliography. For corporate editors, use the corresponding fields `editor` and `shorteditor`. Since all of these fields are treated as name lists, it is possible to mix personal names and corporate names, provided that the names of all corporations and institutions are wrapped in braces.

```
editor      = {{National Aeronautics and Space Administration}
               and Doe, John},
shorteditor = {NASA and Doe, John},
```

Users switching from the `jurabib` package to Biblatex should note that the `shortauthor` field is treated as a name list.

2.3.4 原样输出列表 Literal Lists

The fields `institution`, `organization`, `publisher`, and `location` are literal lists in terms of § 2.2. This also applies to `origlocation`, `origpublisher` and to the field aliases `address` and `school`. All of these fields may contain a list of items separated by the keyword ‘and’. If they contain a literal ‘and’, it must be wrapped in braces.

```
publisher      = {William Reid {and} Company},
institution     = {Office of Information Management {and} Communications},
organization    = {American Society for Photogrammetry {and} Remote Sensing
                  and
                  American Congress on Surveying {and} Mapping},
```

Note the difference between a literal ‘{and}’ and the list separator ‘and’ in the above examples. You may also wrap the entire name in braces:

```
publisher      = {{William Reid and Company}},
institution     = {{Office of Information Management and Communications}},
organization    = {{American Society for Photogrammetry and Remote Sensing}
                  and
                  {American Congress on Surveying and Mapping}},
```

Legacy files which have not been updated for use with Biblatex will still work if these fields do not contain a literal ‘and’. However, note that you will miss out on the additional features of literal lists in this case, such as configurable formatting and automatic truncation.

2.3.5 题名 Titles

The following examples demonstrate how to handle different types of titles. Let’s start with a five-volume work which is referred to as a whole:

```
@MvBook{works,
  author      = {Shakespeare, William},
  title       = {Collected Works},
  volumes     = {5},
  ...}
```

The individual volumes of a multi-volume work usually have a title of their own. Suppose the fourth volume of the *Collected Works* includes Shakespeare’s sonnets and we are referring to this volume only:

```
@Book{works:4,
  author      = {Shakespeare, William},
  maintitle   = {Collected Works},
  title       = {Sonnets},
  volume      = {4},
  ...
}
```

If the individual volumes do not have a title, we put the main title in the `title` field and include a volume number:

```
@Book{works:4,
  author      = {Shakespeare, William},
  title       = {Collected Works},
  volume      = {4},
  ...
}
```

In the next example, we are referring to a part of a volume, but this part is a self-contained work with its own title. The respective volume also has a title and there is still the main title of the entire edition:

```
@InBook{lear,
  author      = {Shakespeare, William},
  bookauthor  = {Shakespeare, William},
  maintitle   = {Collected Works},
  booktitle   = {Tragedies},
  title       = {King Lear},
  volume      = {1},
  pages       = {53-159},
  ...
}
```

Suppose the first volume of the *Collected Works* includes a reprinted essay by a well-known scholar. This is not the usual introduction by the editor but a self-contained work. The *Collected Works* also have a separate editor:

```
@InBook{stage,
  author      = {Expert, Edward},
  title       = {Shakespeare and the Elizabethan Stage},
  bookauthor  = {Shakespeare, William},
  editor      = {Bookmaker, Bernard},
  maintitle   = {Collected Works},
  booktitle   = {Tragedies},
  volume      = {1},
  pages       = {7-49},
  ...
}
```

See § 2.3.7 for further examples.

2.3.6 编辑角色 Editorial Roles

The type of editorial role performed by an editor in one of the `editor` fields (i.e., `editor`, `editora`, `editorb`, `editorc`) may be specified in the corresponding `editor...type` field. The following roles are supported by default. The role ‘`editor`’ is the default. In this case, the `editortype` field is omissible.

<code>editor</code>	The main editor. This is the most generic editorial role and the default value.
<code>compiler</code>	Similar to <code>editor</code> but used if the task of the editor is mainly compiling.
<code>founder</code>	The founding editor of a periodical or a comprehensive publication project such as a ‘Collected Works’ edition or a long-running legal commentary.
<code>continuator</code>	An editor who continued the work of the founding editor (<code>founder</code>) but was subsequently replaced by the current editor (<code>editor</code>).
<code>redactor</code>	A secondary editor whose task is redacting the work.
<code>reviser</code>	A secondary editor whose task is revising the work.
<code>collaborator</code>	A secondary editor or a consultant to the editor.

For example, if the task of the editor is compiling, you may indicate that in the corresponding `editortype` field:

```
@Collection{...,
  editor      = {Editor, Edward},
  editortype  = {compiler},
  ...
```

There may also be secondary editors in addition to the main editor:

```
@Book{...,
  author      = {...},
  editor      = {Editor, Edward},
  editora     = {Redactor, Randolph},
  editoratype = {redactor},
  editorb     = {Consultant, Conrad},
  editorbtype = {collaborator},
  ...
```

Periodicals or long-running publication projects may see several generations of editors. For example, there may be a founding editor in addition to the current editor:

```
@Book{...,
  author      = {...},
  editor      = {Editor, Edward},
  editora     = {Founder, Frederic},
  editoratype = {founder},
  ...
```


Note that only the `editor` is considered in citations and when sorting the bibliography. If an entry is typically cited by the founding editor (and sorted accordingly in the bibliography), the founder goes into the `editor` field and the current editor moves to one of the `editor...` fields:

```
@Collection{...,
  editor      = {Founder, Frederic},
  editortype  = {founder},
  editora     = {Editor, Edward},
  ...}
```

You may add more roles by initializing and defining a new localization key whose name corresponds to the identifier in the `editor...type` field. See §§ 3.8 and 4.9.1 for details.

2.3.7 出版物和期刊系列 **Publication and Journal Series**

The `series` field is used by traditional BibTeX styles both for the main title of a multi-volume work and for a publication series, i.e., a loosely related sequence of books by the same publisher which deal with the same general topic or belong to the same field of research. This may be ambiguous. This package introduces a `maintitle` field for multi-volume works and employs `series` for publication series only. The volume or number of a book in the series goes in the `number` field in this case:

```
@Book{...,
  author      = {Expert, Edward},
  title       = {Shakespeare and the Elizabethan Age},
  series      = {Studies in English Literature and Drama},
  number      = {57},
  ...}
```

The `@article` entry type makes use of the `series` field as well, but handles it in a special way. First, a test is performed to determine whether the value of the field is an integer. If so, it will be printed as an ordinal. If not, another test is performed to determine whether it is a localization key. If so, the localized string is printed. If not, the value is printed as is. Consider the following example of a journal published in numbered series:

```
@Article{...,
  journal     = {Journal Name},
  series      = {3},
  volume      = {15},
  number      = {7},
  year        = {1995},
  ...}
```

Date Specification	Formatted Date (Examples)	
	Short Format	Long Format
1850	1850	1850
1997/	1997–	1997–
1967-02	02/1967	February 1967
2009-01-31	31/01/2009	31st January 2009
1988/1992	1988–1992	1988–1992
2002-01/2002-02	01/2002–02/2002	January 2002–February 2002
1995-03-30/1995-04-05	30/03/1995–05/04/1995	30th March 1995–5th April 1995

Table 3: Date Specifications

This entry will be printed as “*Journal Name*. 3rd ser. 15.7 (1995)”. Some journals use designations such as “old series” and “new series” instead of a number. Such designations may be given in the `series` field as well, either as a literal string or as a localization key. Consider the following example which makes use of the localization key `newseries`:

```
@Article{...,
  journal      = {Journal Name},
  series       = {newseries},
  volume       = {9},
  year         = {1998},
  ...}
```

This entry will be printed as “*Journal Name*. New ser. 9 (1998)”. See § 4.9.2 for a list of localization keys defined by default.

2.3.8 日期格式 Date Specifications

The date fields `date`, `origdate`, `eventdate`, and `urldate` require a date specification in `yyyy-mm-dd` format. Date ranges are given as `yyyy-mm-dd/yyyy-mm-dd`. Partial dates are valid provided that date components are omitted at the end only. You may specify an open ended date range by giving the range separator and omitting the end date (e. g., `yyyy/`). See table 3 for some examples of valid date specifications and the formatted date automatically generated by Biblatex. The formatted date is language specific and will be adapted automatically. If there is no date field in an entry, Biblatex will also consider the fields `year` and `month` for backwards compatibility with traditional BibTeX. Style author should note that date fields like `date` or `origdate` are only available in the `bib` file. All dates are parsed and dissected into their components as the `bib` file is processed. The date components are made available to styles by way of the special fields discussed in § 4.2.4.3. See this section and table 8 on page 157 for further information.

2.3.9 月份和期刊季度号 Months and Journal Issues

The month field is an integer field. The bibliography style converts the month to a language-dependent string as required. For backwards compatibility, you may also use the following three-letter abbreviations in the month field: jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec. Note that these abbreviations are BibTeX strings which must be given without any braces or quotes. When using them, don't say `month={jan}` or `month="jan"` but `month=jan`. It is not possible to specify a month such as `month={8/9}`. Use the date field for date ranges instead.

用于区分季刊的‘Spring’或‘Summer’应该放在issue域内，@article条目的issue域位置类似于month域并覆盖该域。

2.3.10 标记页码 Pagination

当在条目的pages域或者一个引用命令的⟨*postnote*⟩参数中指定页码或者页码范围，很方便让 Biblatex 自动添加像 ‘p.’ 或 ‘pp.’ 之类的前缀，这也是该包的默认处理方式。然而，一些文档可能使用一种不同的页码标记格式或者可能不是以页码而是以诗节号或者行号为标记。这时pagination和bookpagination域就可以起到作用了，举例考虑下面的条目：

```
@InBook{key,
  title           = {...},
  pagination      = {verse},
  booktitle       = {...},
  bookpagination = {page},
  pages           = {53--65},
  ...
```

The bookpagination field affects the formatting of the pages and pagetotal fields in the list of references. Since page is the default, this field is omissible in the above example. In this case, the page range will be formatted as ‘pp. 53–65’. Suppose that, when quoting from this work, it is customary to use verse numbers rather than page numbers in citations. This is reflected by the pagination field, which affects the formatting of the ⟨*postnote*⟩ argument to any citation command. With a citation like `\cite[17]{key}`, the postnote will be formatted as ‘v. 17’. Setting the pagination field to section would yield ‘§ 17’. See § 3.12.3 for further usage instructions.

The pagination and bookpagination fields are key fields. This package will try to use their value as a localization key, provided that the key is defined. Always use the singular form of the key name in bib files, the plural is formed automatically. The keys page, column, line, verse, section, and paragraph are predefined, with page being the default. The string ‘none’ has a special meaning when used in a pagination or bookpagination field. It suppresses the prefix for the respective entry. If there are no predefined localization keys for the pagination scheme required by a certain entry, you can simply add them. See the com-

mands `\NewBibliographyString` and `\DefineBibliographyStrings` in § 3.8. You need to define two localization strings for each additional pagination scheme: the singular form (whose localization key corresponds to the value of the pagination field) and the plural form (whose localization key must be the singular plus the letter ‘s’). See the predefined keys in § 4.9.2 for examples.

2.4 提示与警告 Hints and Caveats

This section provides some additional hints concerning the data interface of this package. It also addresses some common problems.

2.4.1 交叉引用 Cross-referencing

2.4.1.1 The `crossref` field (BibTeX) The `crossref` field is a convenient way to establish a parent/child relation between two associated entries. Unfortunately, the BibTeX program uses symmetric field mapping which reduces the usefulness of the `crossref` field significantly. There are two issues with symmetric field mapping, as seen in the following example:

```
@Book{book,
  author      = {Author},
  bookauthor  = {Author},
  title       = {Booktitle},
  booktitle   = {Booktitle},
  subtitle    = {Booksubtitle},
  booksubtitle = {Booksubtitle},
  publisher   = {Publisher},
  location    = {Location},
  date        = {1995},
}
@InBook{inbook,
  crossref    = {book},
  title       = {Title},
  subtitle    = {},
  pages       = {5--25},
}
```

As BibTeX is not capable of mapping the `title` field of the parent to the `booktitle` field of the child, the title of the book needs to be given twice. The style then needs to ignore the `booktitle` of the parent since it is only required to work around this fundamental limitation of BibTeX. The problem with the `subtitle` field is the inverse of that. Since the `subtitle` of the parent would become the `subtitle`, rather than in the `booksubtitle`, of the child, we need to add an empty `subtitle` field to the child entry to prevent inheritance of this field. Of course we also need to duplicate the subtitle in the parent entry to ensure that it is available as `booksubtitle` in the

child entry. In short, using BibTeX's `crossref` field tends to bloat database files and corrupt the data model.

2.4.1.2 The `crossref` field (Biber) With Biber, the limitations of BibTeX's `crossref` field belong to the past. Biber features a highly customizable cross-referencing mechanism with flexible data inheritance rules. Duplicating certain fields in the parent entry or adding empty fields to the child entry is no longer required. Entries are specified in a natural way:

```
@Book{book,
  author      = {Author},
  title       = {Booktitle},
  subtitle    = {Booksubtitle},
  publisher   = {Publisher},
  location    = {Location},
  date        = {1995},
}
@InBook{inbook,
  crossref    = {book},
  title       = {Title},
  pages       = {5--25},
}
```

The `title` field of the parent will be copied to the `booktitle` field of the child, the `subtitle` becomes the `booksubtitle`. The `author` of the parent becomes the `bookauthor` of the child and, since the child does not provide an `author` field, it is also duplicated as the `author` of the child. After data inheritance, the child entry is similar to this:

```
author      = {Author},
bookauthor  = {Author},
title       = {Title},
booktitle   = {Booktitle},
booksubtitle = {Booksubtitle},
publisher   = {Publisher},
location    = {Location},
date        = {1995},
pages       = {5--25},
```

See appendix B for a list of mapping rules set up by default. Note that all of this is customizable. See § 4.5.10 on how to configure Biber's cross-referencing mechanism. See also § 2.2.3.

Parameter	Switch	Capacity			
		Default	--big	--huge	--wolfgang
max_cites	--mcites	750	2000	5000	7500
max_ent_ints	--mentints	3000	4000	5000	7500
max_ent_strs	--mentstrs	3000	6000	10000	10000
max_fields	--mfields	17250	30000	85000	125000
max_strings	--mstrings	4000	10000	19000	30000
pool_size	--mpool	65530	130000	500000	750000
wiz_fn_space	--mwizfuns	3000	6000	10000	10000
hash_prime		4253	8501	16319	30011
hash_size		5000	10000	19000	35000

Table 4: Capacity and Switches of `bibtex8`

2.4.1.3 The `xref` field In addition to the `crossref` field, Biblatex supports a simplified cross-referencing mechanism based on the `xref` field. This is useful if you want to establish a parent/child relation between two associated entries but prefer to keep them independent as far as the data is concerned. The `xref` field differs from `crossref` in that the child entry will not inherit any data from the parent. If the parent is referenced by a certain number of child entries, Biblatex will automatically add it to the bibliography. The threshold is controlled by the `mincrossrefs` package option from § 3.1.2.1. The `xref` field is supported with all backends. See also § 2.2.3.

2.4.2 能力问题 Capacity Issues

2.4.2.1 BibTeX 当参考文献量较大时, BibTeX 可能存在内存不足的情形, 这里给出了一些出错情况。最后建议使用 `bibtex8` 或 `biber`。

2.4.2.2 `bibtex8` `bibtex8` 比之 `bibtex` 有所增强, 还有一些参数设置可以增大默认能力。但其能力仍然有限, 还可能存在内存不足的问题。

2.4.2.3 Biber `Biber` 可以消除上述所有局限。

2.4.3 排序和编码问题 Sorting and Encoding Issues

2.4.3.1 BibTeX Traditional BibTeX can only alphabetize Ascii characters correctly. If the bibliographic data includes non-Ascii characters, they have to be given in Ascii notation. For example, instead of typing a letter like ‘ä’ directly, you need to input it as `\“a`, using an accent command and the Ascii letter. This Ascii notation needs to be wrapped in a pair of curly braces. Traditional BibTeX will then ignore the accent and use the Ascii letter for sorting. Here are a few examples:

```
author      = {S{\“a}nchez, Jos{\“e}},
editor      = {Ma{\ss}mann, R{\“u}diger},
translator  = {Ferd{\e}re, Fr{\c{c}}ois},
title       = {{\OE}uvres compl{\e}tes},
```

Apart from it being inconvenient, there are two major issues with this convention. One subtle problem is that the extra set of braces suppresses the kerning on both sides of all non-Ascii letters. But first and foremost, simply ignoring all accents may not be the correct way to handle them. For example, in Danish, the letter ‘å’ is the very last letter of the alphabet, so it should be alphabetized after ‘z’. BibTeX will sort it like an ‘a’. The ‘æ’ ligature and the letter ‘ø’ are also sorted after ‘z’ in this language. There are similar cases in Norwegian. In Swedish, the letter ‘ö’ is the very last letter of the alphabet and the letters ‘å’ and ‘ä’ are also alphabetized after ‘z’, rather than like an ‘a’. What’s more, even the sorting of Ascii characters is done in a rather peculiar way by traditional BibTeX because the sorting algorithm uses Ascii codepage order (0–9, A–Z, a–z). This implies that the lowercase letter ‘a’ would end up after the uppercase ‘Z’, which is not even acceptable in the language BibTeX was originally designed for. The traditional `bst` files work around this problem by converting all strings used for sorting to lowercase, i. e., sorting is effectively case-insensitive. See also § 2.4.3.4.

2.4.3.2 `bibtex8` Switching to `bibtex8` will help in such cases. `bibtex8` can sort case-sensitively and it can handle 8-bit characters properly, provided that you supply it with a suitable `csf` file and give the `--csfile` switch on the command line. This also implies that it is possible to apply language specific sorting rules to the bibliography. The Biblatex package comes with `csf` files for some common Western European encodings. `bibtex8` also ships with a few `csf` files. Note that `biblatex.bst` can not detect if it is running under traditional BibTeX or `bibtex8`, hence the `bibtex8` package option. By default, sorting is case-insensitive since this is required for traditional BibTeX. If the `bibtex8` package option is enabled, sorting is case-sensitive.

Since `bibtex8` is backwards compatible with traditional BibTeX, it is possible to mix 8-bit input and Ascii notation. This is useful if the encoding used in the `bib` file does not cover all required characters. There are also a few marginal cases in which the Ascii notation scheme would yield better sorting results. A typical example is the ligature ‘œ’. `bibtex8` will handle this ligature like a single character. Depending on the sorting scheme defined in the `csf` file, it could be treated like an ‘o’ or alphabetized after the letter ‘o’ but it can not be sorted as ‘oe’. The Ascii notation (`\oe`) is equivalent to ‘œ’ during sorting:

```
title      = {\oeuvres complètes},
title      = {\OEuvres complètes},
```

Sometimes even that is not sufficient and further tricks are required. For example, the letter ‘ß’ in German is particularly tricky. This letter is essentially alphabetized as ‘ss’ but after ‘ss’. The name ‘Baßmann’ would be alphabetized as follows: Bassmann/Bassmann/Baßmann/Bastmann. In this case, the Ascii notation (`\ss`) would yield slightly better sorting results than ‘ß’ in conjunction with a `csf` file which treats ‘ß’ like ‘s’:

```
author     = {Ba{\ss}mann, Paul},
```

To get it absolutely right, however, you need to resort to the `sortname` field:

```
author      = {Baßmann, Paul},  
sortname    = {Basszmann, Paul},
```

Not only BibTeX, LaTeX needs to know about the encoding as well. See § 2.4.3.4 on how to specify encodings.

2.4.3.3 Biber Biber handles Ascii, 8-bit encodings such as Latin 1, and UTF-8. It features true Unicode support and is capable of reencoding the `bib` data on the fly in a robust way. For sorting, Biber uses a Perl implementation of the Unicode Collation Algorithm (UCA), as outlined in Unicode Technical Standard #10.⁸ Collation tailoring based on the Unicode Common Locale Data Repository (CLDR) is also supported.⁹ The bottom line is that Biber will deliver sorting results far superior to both BibTeX and `bibtex8` in many cases. If you are interested in the technical details, section 1.8 of Unicode Technical Standard #10 will provide you with a very concise summary of why the inadequateness of traditional BibTeX and even `bibtex8` is of a very general nature and not limited to the lack of UTF-8 support.¹⁰

Supporting Unicode implies much more than handling UTF-8 input. Unicode is a complex standard covering more than its most well-known parts, the Unicode character encoding and transport encodings such as UTF-8. It also standardizes aspects such as string collation, which is required for language-sensitive sorting. For example, by using the Unicode Collation Algorithm, Biber can handle the character ‘ß’ mentioned as an example in § 2.4.3.2 without any manual intervention. All you need to do to get localized sorting is specify the locale:

```
\usepackage[backend=biber,sortlocale=de]{biblatex}
```

or if you are using german as the main document language via Babel or Polyglossia:

```
\usepackage[backend=biber,sortlocale=auto]{biblatex}
```

This will make Biblatex pass the Babel/Polyglossia main document language as the locale which Biber will map into a suitable default locale. Biber will not try to get locale information from its environment as this makes document processing dependent on something not in the document which is against TeX’s spirit of reproducibility. This also makes sense since Babel/Polyglossia are in fact the relevant environment for a document. Note that this will also work with 8-bit encodings such as Latin 9, i. e., you can take advantage of Unicode-based sorting even though you are not using UTF-8 input. See § 2.4.3.4 on how to specify input and data encodings properly.

⁸<http://unicode.org/reports/tr10/>

⁹<http://cldr.unicode.org/>

¹⁰http://unicode.org/reports/tr10/#Common_Misperceptions

2.4.3.4 Specifying Encodings When using a non-Ascii encoding in the `bib` file, it is important to understand what Bibl_{at}ex can do for you and what may require manual intervention. The package takes care of the LaTeX side, i.e., it ensures that the data imported from the `bbl` file is interpreted correctly, provided that the `bibencoding` package option is set properly. Depending on the backend, the BibTeX side may demand attention, too. When using `bibtex8`, you need to supply `bibtex8` with a matching `csf` file as it needs to know about the encoding of the `bib` file to be able to alphabetize the entries correctly. Unfortunately, there is no way for Bibl_{at}ex to pass this information to `bibtex8` automatically. The only way is setting its `--csfile` option on the command line when running `bibtex8`. When using Biber, all of this is handled automatically and no further steps, apart from setting the `bibencoding` option in certain cases, are required. Here are a few typical usage scenarios along with the relevant lines from the document preamble:

- Ascii notation in both the `tex` and the `bib` file with pdfTeX or traditional TeX (this will work with BibTeX, `bibtex8`, and Biber):

```
\usepackage{biblatex}
```

- Latin 1 encoding (iso-8859-1) in the `tex` file, Ascii notation in the `bib` file with pdfTeX or traditional TeX (BibTeX, `bibtex8`, Biber):

```
\usepackage[latin1]{inputenc}
\usepackage[bibencoding=ascii]{biblatex}
```

- Latin 9 encoding (iso-8859-15) in both the `tex` and the `bib` file with pdfTeX or traditional TeX (`bibtex8`, Biber):

```
\usepackage[latin9]{inputenc}
\usepackage[bibencoding=auto]{biblatex}
```

Since `bibencoding=auto` is the default setting, the option is omissible. The following setup will have the same effect:

```
\usepackage[latin9]{inputenc}
\usepackage{biblatex}
```

- UTF-8 encoding in the `tex` file, Latin 1 (iso-8859-1) in the `bib` file with pdfTeX or traditional TeX (`bibtex8`, Biber):

```
\usepackage[utf8]{inputenc}
\usepackage[bibencoding=latin1]{biblatex}
```

The same scenario with XeTeX or LuaTeX in native UTF-8 mode:

```
\usepackage[bibencoding=latin1]{biblatex}
```

- Using UTF-8 encoding in both the `tex` and the `bib` file is not possible with traditional BibTeX or `bibtex8` since neither of them is capable of handling UTF-8. Unless you switch to Biber, you need to use an 8-bit encoding such as Latin 1 (see above) or resort to Ascii notation in this case:

```
\usepackage[utf8]{inputenc}
\usepackage[bibencoding=ascii]{biblatex}
```

The same scenario with XeTeX or LuaTeX in native UTF-8 mode:

```
\usepackage[bibencoding=ascii]{biblatex}
```

Biber can handle Ascii notation, 8-bit encodings such as Latin 1, and UTF-8. It is also capable of reencoding the `bib` data on the fly (replacing the limited macro-level reencoding feature of Biblatex). This will happen automatically if required, provided that you specify the encoding of the `bib` files properly. In addition to the scenarios discussed above, Biber can also handle the following cases:

- Transparent UTF-8 workflow, i. e., UTF-8 encoding in both the `tex` and the `bib` file with pdfTeX or traditional TeX:

```
\usepackage[utf8]{inputenc}
\usepackage[bibencoding=auto]{biblatex}
```

Since `bibencoding=auto` is the default setting, the option is omissible:

```
\usepackage[utf8]{inputenc}
\usepackage{biblatex}
```

The same scenario with XeTeX or LuaTeX in native UTF-8 mode:

```
\usepackage{biblatex}
```

- It is even possible to combine an 8-bit encoded `tex` file with UTF-8 encoding in the `bib` file, provided that all characters in the `bib` file are also covered by the selected 8-bit encoding:

```
\usepackage[latin1]{inputenc}
\usepackage[bibencoding=utf8]{biblatex}
```

Some workarounds may be required when using traditional TeX or pdfTeX with UTF-8 encoding because `inputenc`'s `utf8` module does not cover all of Unicode. Roughly speaking, it only covers the Western European Unicode range. When loading `inputenc` with the `utf8` option, Biblatex will normally instruct Biber to reencode the `bib` data to UTF-8. This may lead to `inputenc` errors if some of the characters in the `bib` file are outside the limited Unicode range supported by `inputenc`.

- If you are affected by this problem, try setting the `safeinputenc` option:

```
\usepackage[utf8]{inputenc}
\usepackage[safeinputenc]{biblatex}
```

If this option is enabled, Biblatex will ignore `inputenc`'s `utf8` option and use `Ascii`. Biber will then try to convert the `bib` data to `Ascii` notation. For example, it will convert `Si` to `\k{S}`. This option is similar to setting `texencoding=ascii` but will only take effect in this specific scenario (`inputenc/inputenx` with UTF-8). This workaround takes advantage of the fact that both Unicode and the UTF-8 transport encoding are backwards compatible with `Ascii`.

This solution may be acceptable as a workaround if the data in the `bib` file is mostly `Ascii` anyway, with only a few strings, such as some authors' names, causing problems. However, keep in mind that it will not magically make traditional TeX or pdfTeX support Unicode. It may help if the occasional odd character is not supported by `inputenc`, but may still be processed by TeX when using an accent command (e. g., `\d{S}` instead of `§`). If you need full Unicode support, however, switch to XeTeX or LuaTeX.

Typical errors when `inputenc` cannot handle a certain UTF-8 character are:

```
Package inputenc Error: Unicode char \u8: not set up for use with LaTeX
```

but also less obvious things like:

```
! Argument of \UTFviii@three@octets has an extra }.
```

2.4.4 编纂者脚本 Editors and Compiler Scripts

本节需要更新以配合 Biblatex 使用的新的脚本接口。目前，草稿内容可以参考 `logreq` 包的说明文档¹¹ 和 Biblatex 开发者百科¹²。

3 用户使用手册

本部分介绍 `biblatex` 包的用户接口。这一部分的用户指南包含了所有要使用标准样式的信息。无论如何首先要阅读这一用户指南，如果要写自己的样式，则需要继续阅读后面的作者指南。

¹¹<http://www.ctan.org/tex-archive/macros/latex/contrib/logreq/>

¹²http://sourceforge.net/apps/mediawiki/biblatex/index.php?title=Workflow_Automation

3.1 宏包选项

所有的包选项都以 $\langle key \rangle = \langle value \rangle$ 形式标记。对于所有的布尔选项，其值 `true` 都可以忽略。比如给出`sortcites`不带选项值等价于 `sortcites=true`。

3.1.1 加载选项

下面的选项必须在 `Biblatex` 加载时给出，即作为`\usepackage`命令的可选参数。

`backend=bibtex, bibtex8, bibtexu, biber` default: biber

用于指定数据库后端。支持的后端包括:

- | | |
|----------------------|--|
| <code>biber</code> | <code>Biber</code> , <code>Biblatex</code> 的默认后端, 支持 <code>Ascii</code> , <code>8-bit</code> 编码, <code>UTF-8</code> , 再编码 ¹³ , 本地定制排序和很多其他特性。本地定制排序、大小写敏感排序、大小写优先分别由 <code>sortlocale</code> , <code>sortcase</code> , 和 <code>sortupper</code> 选项控制。 |
| <code>bibtex</code> | 遗留的 <code>BibTeX</code> . 传统的 <code>BibTeX</code> 仅支持 <code>Ascii</code> 编码。排序总是大小写敏感。 |
| <code>bibtex8</code> | <code>bibtex8</code> , <code>BibTeX</code> 的 <code>8-bit</code> 实现, 支持 <code>Ascii</code> 和 <code>8-bit</code> 编码比如 <code>Latin 1</code> 。根据 <code>csf</code> 文件, 可以支持大小写敏感排序 |
| <code>bibtexu</code> | <code>bibtexu</code> 是 <code>BibTeX</code> 为支持 <code>Unicode</code> 的实现版本, 可以支持 <code>UTF-8</code> 。注意 <code>Biblatex</code> 不主动支持 <code>bibtexu</code> , 并没有以任何方式进行测试。 <code>Biber</code> 则是推荐的后端。 |

关于`bib`文件编码的更多说明见 § 2.4.3 节。该选项仅在加载的是偶作为内部选项提供, 代码根据 `backend` 选择完全不同的路径。这意味着不能用导言区的命令比如`\ExecuteBibliographyOptions`设置后端。

`style=\langle file \rangle` default: numeric

加载参考文献著录样式`file.bbx` 和标注样式`file.cbx`。各标准样式介绍见 § 3.3 节。

`bibstyle=\langle file \rangle` default: numeric

加载参考文献著录样式`file.bbx`。各标准著录样式介绍见 § 3.3.2 节。

`citestyle=\langle file \rangle` default: numeric

加载参考文献标注样式`file.cbx`。各标准标注样式介绍见 § 3.3.1 节。

`natbib=true, false` default: false

加载兼容模块, 提供了`natbib`包的引用命令的同名替代命令。详见 § 3.7.9 节。

`mcite=true, false` default: false

加载一个引用命令模块, 提供了类似`mcite/mciteplus`的引用命令。详见 § 3.7.10 节。

¹³on-the-fly?

3.1.2 导言区选项

3.1.2.1 一般选项 下面的选项可以作为`\usepackage`命令的可选参数，也可以在配置文件和导言区设置。默认值列在右侧作为包的默认值。注意著录和标注样式可以修改加载时的默认设置，详见 § 3.3 节。

`sorting=nty, nyt, nyvt, anyt, anyvt, ynt, ydnt, none, debug,` default: `nty`
`<name>`

参考文献的排序方式。除非进行声明，否则条目以升序排列。下面提供的是预设可选值：

`nty` Sort by name, title, year.

`nyt` Sort by name, year, title.

`nyvt` Sort by name, year, volume, title.

`anyt` Sort by alphabetic label, name, year, title.

`anyvt` Sort by alphabetic label, name, year, volume, title.

`ynt` Sort by year, name, title.

`ydnt` Sort by year (descending), name, title.

`none` Do not sort at all. All entries are processed in citation order.

`debug` Sort by entry key. This is intended for debugging only.

`<name>` Use `<name>`, as defined with `\DeclareSortingScheme` (§ 4.5.5) Biber only

Using any of the ‘alphabetic’ sorting schemes only makes sense in conjunction with a bibliography style which prints the corresponding labels. Note that some bibliography styles initialize this package option to a value different from the package default (`nty`). See § 3.3.2 for details. Please refer to § 3.5 for an in-depth explanation of the above sorting options as well as the fields considered in the sorting process. See also § 4.5.5 on how to adapt the predefined schemes or define new ones.

`sortcase=true, false` default: `true`

Whether or not to sort the bibliography and the list of shorthands case-sensitively. Note that case-sensitive sorting is only supported by the `bibtex8` and Biber backends. Sorting is always case-insensitive with legacy BibTeX. See the `backend` option for details.

`sortupper=true, false` default: `true` Biber only

This option corresponds to Biber’s `--sortupper` command-line option. It has no effect with any other backend. If enabled, the bibliography is sorted in ‘uppercase before lowercase’ order. Disabling this option means ‘lowercase before uppercase’ order.

`sortlocale=auto, <locale>` Biber only

This option sets the global sorting locale. Every sorting scheme inherits this locale if none is specified using the `<locale>` option to `\printbibliography`. Setting this to `auto` requests that it be set to the Babel/Polyglossia main document language identifier, if these packages are used and `en_US` otherwise. Biber will map Babel/Polyglossia language identifiers into sensible locale identifiers (see the Biber documentation). You can therefore specify either a normal locale identifier like `de_DE_phonebook`, `es_ES` or one of the supported Babel/Polyglossia language identifiers if the mapping Biber makes of this is fine for you.

`sortlos=bib, los` default: `los` BibTeX only

The sorting order of the list of shorthands. The following choices are available:

`bib` Sort according to the sorting order of the bibliography.
`los` Sort by shorthand.

The sorting order of shorthands with Biber is more flexible and is set with the `sorting` option to the `\printbiblist` command.

`related=true, false` default: `true` Biber only

Whether or not to use information from related entries or not. See § 3.4.

是否使用相关条目的信息。

`sortcites=true, false` default: `false`

Whether or not to sort citations if multiple entry keys are passed to a citation command. If this option is enabled, citations are sorted according to the current bibliography context sorting scheme (see § 3.6.11). This feature works with all citation styles.

当多个条目关键词传给引用命令时，是否进行排序。

`maxnames=<integer>` default: `3`

A threshold affecting all lists of names (author, editor, etc.). If a list exceeds this threshold, i.e., if it holds more than `<integer>` names, it is automatically truncated according to the setting of the `minnames` option. `maxnames` is the master option which sets both `maxbibnames` and `maxcitenames`.

最大名字数量，超过截断。

`minnames=<integer>` default: `1`

A limit affecting all lists of names (author, editor, etc.). If a list holds more than `<maxnames>` names, it is automatically truncated to `<minnames>` names. The `<minnames>` value must be smaller than or equal to `<maxnames>`. `minnames` is the master option which sets both `minbibnames` and `mincitenames`.

超过多于`<maxnames>`的名字，截断到`<minnames>` names。

`maxbibnames`= $\langle integer \rangle$ default: $\langle maxnames \rangle$

Similar to `maxnames` but affects only the bibliography.

`minbibnames`= $\langle integer \rangle$ default: $\langle minnames \rangle$

Similar to `minnames` but affects only the bibliography.

`maxcitenames`= $\langle integer \rangle$ default: $\langle maxnames \rangle$

Similar to `maxnames` but affects only the citations in the document body.

`mincitenames`= $\langle integer \rangle$ default: $\langle minnames \rangle$

Similar to `minnames` but affects only the citations in the document body.

`maxitems`= $\langle integer \rangle$ default: 3

Similar to `maxnames`, but affecting all literal lists (publisher, location, etc.).

最大项数类似于姓名的处理，用于publisher, location等域。

`minitems`= $\langle integer \rangle$ default: 1

Similar to `minnames`, but affecting all literal lists (publisher, location, etc.).

`autocite`=plain, inline, footnote, superscript, ...

控制引用文字的样式。注意 `superscript` 选项只有在以数字引用的样式中使用，其它则总是可以。

This option controls the behavior of the `\autocite` command discussed in § 3.7.4. The `plain` option makes `\autocite` behave like `\cite`, `inline` makes it behave like `\parencite`, `footnote` makes it behave like `\footcite`, and `superscript` makes it behave like `\supercite`. The options `plain`, `inline`, and `footnote` are always available, the `superscript` option is only provided by the numeric citation styles which come with this package. The citation style may also define additional options. The default setting of this option depends on the selected citation style, see § 3.3.1.

`autopunct`=true, false default: true

This option controls whether the citation commands scan ahead for punctuation marks. See § 3.7 and `\DeclareAutoPunctuation` in § 4.7.5 for details.

控制引用命令是否在标点前扫描。

`language`=autobib, autocite, auto, $\langle language \rangle$ default: autobib

This option controls multilingual support. When set to `autobib`, `autocite` or `auto`, Biblatex will try to get the main document language from the `babel/polyglossia` package (and fall back to English if `babel/polyglossia` is not available). It is also possible to select the document language manually. In this

case, the `autolang` option below will have no effect. Please refer to table 2 for a list of supported languages and the corresponding identifiers. `autobib` switches the language for each entry in the bibliography using the `langid` field and the language environment specified by the `autolang` option. `autocite` switches the language for each citation using the `langid` field and the language environment specified by the `autolang` option. `auto` is a shorthand to set both `autobib` and `autocite`. The default is to switch languages automatically only for bibliography entries (`autobib`).

`clearlang=true, false` default: true

If this option is enabled, Biblatex will automatically clear the `language` field of all entries whose language matches the `babel/polyglossia` language of the document (or the language specified explicitly with the `language` option) in order to omit redundant language specifications. The language mappings required by this feature are provided by the `\DeclareRedundantLanguages` command from § 4.9.1.

`autolang=none, hyphen, other, other*, langname` default: none

This option controls which `babel` language environment¹⁴ is used if the `babel/polyglossia` package is loaded and a bibliography entry includes a `langid` field (see § 2.2.3). Note that Biblatex automatically adjusts to the main document language if `babel/polyglossia` is loaded. In multilingual documents, it will also continually adjust to the current language as far as citations and the default language of the bibliography is concerned. This option is for switching languages on a per-entry basis within the bibliography. The possible choices are:

- `none` Disable this feature, i. e., do not use any language environment at all.
- `hyphen` Enclose the entry in a `hyphenrules` environment. This will load hyphenation patterns for the language specified in the `hyphenation` field of the entry, if available.
- `other` Enclose the entry in an `otherlanguage` environment. This will load hyphenation patterns for the specified language, enable all extra definitions which `babel/polyglossia` and Biblatex provide for the respective language, and translate key terms such as ‘editor’ and ‘volume’. The extra definitions include localizations of the date format, of ordinals, and similar things.
- `other*` Enclose the entry in an `otherlanguage*` environment. Please note that Biblatex treats `otherlanguage*` like `otherlanguage` but other packages may make a distinction in this case.
- `langname` `polyglossia` only. Enclose the entry in a ‘`language`name’ environment. The benefit of this option value for `polyglossia` users

¹⁴`polyglossia` understands the `babel` language environments too and so this option controls both the `babel` and `polyglossia` language environments.

is that it takes note of the `langidopts` field so that you can add per-language options to an entry (like selecting a language variant). When using `babel`, this option does the same as the `other` option value.

`block=none, space, par, nbpar, ragged` default: none

`block=` 选项控制块之间的额外空间，比如参考文献条目的更大部分。

This option controls the extra spacing between blocks, i. e., larger segments of a bibliography entry. The possible choices are:

- | | |
|---------------------|--|
| <code>none</code> | Do not add anything at all. 不添加任何东西 |
| <code>space</code> | Insert additional horizontal space between blocks. This is similar to the default behavior of the standard LaTeX document classes. 在块之间插入水平空间，类似于 latex 文档类的默认方式。 |
| <code>par</code> | Start a new paragraph for every block. This is similar to the <code>openbib</code> option of the standard LaTeX document classes. 开启新的一段，类似于标准 latex 文档类的 <code>openbib</code> 选项。 |
| <code>nbpar</code> | Similar to the <code>par</code> option, but disallows page breaks at block boundaries and within an entry. 类似域 <code>par</code> 选项，但在条目中块边界处不允许分页。 |
| <code>ragged</code> | Inserts a small negative penalty to encourage line breaks at block boundaries and sets the bibliography ragged right. 插入一个负的罚值用于鼓励在块的编辑处换行，使其左对齐。 |

The `\newblockpunct` command may also be redefined directly to achieve different results, see § 3.9.1. Also see § 4.7.1 for additional information.

`\newblockpunct` 可以重定义实现不同的效果，见 § 3.9.1。更多的信息参见 § 4.7.1。

`notetype=foot+end, footonly, endonly` default: foot+end

该选项控制 `\mkbibfootnote`, `\mkbibendnote` 或类似 § 4.10.4 的 wrappers 的行为。

This option controls the behavior of `\mkbibfootnote`, `\mkbibendnote`, and similar wrappers from § 4.10.4. The possible choices are:

- | | |
|-----------------------|---|
| <code>foot+end</code> | Support both footnotes and endnotes, i. e., <code>\mkbibfootnote</code> will generate footnotes and <code>\mkbibendnote</code> will generate endnotes.
脚注和尾注分开 |
| <code>footonly</code> | Force footnotes, i. e., make <code>\mkbibendnote</code> generate footnotes.
都生成脚注 |
| <code>endonly</code> | Force endnotes, i. e., make <code>\mkbibfootnote</code> generate endnotes.
都生成尾注 |

`hyperref=true, false, auto`

default: `auto`

是否形成超连接，需要样式文件支持。标准样式支持。

Whether or not to transform citations and back references into clickable hyperlinks. This feature requires the `hyperref` package. It also requires support by the selected citation style. All standard styles which ship with this package support hyperlinks. `hyperref=auto` automatically detects if the `hyperref` package has been loaded.

`backref=true, false`

default: `false`

是否形成反向超连接，需要样式文件支持。标准样式支持。

Whether or not to print back references in the bibliography. The back references are a list of page numbers indicating the pages on which the respective bibliography entry is cited. If there are `refsection` environments in the document, the back references are local to the reference sections. Strictly speaking, this option only controls whether the Biblatex package collects the data required to print such references. This feature still has to be supported by the selected bibliography style. All standard styles which ship with this package do so.

`backrefstyle=none, three, two, two+, three+, all+`

default: `three`

控制反向连接的页码连续形式。

This option controls how sequences of consecutive pages in the list of back references are formatted. The following styles are available:

- | | |
|---------------------|---|
| <code>none</code> | Disable this feature, i. e., do not compress the page list. |
| <code>three</code> | Compress any sequence of three or more consecutive pages to a range, e. g., the list ‘1, 2, 11, 12, 13, 21, 22, 23, 24’ is compressed to ‘1, 2, 11–13, 21–24’. |
| <code>two</code> | Compress any sequence of two or more consecutive pages to a range, e. g., the above list is compressed to ‘1–2, 11–13, 21–24’. |
| <code>two+</code> | Similar in concept to <code>two</code> but a sequence of exactly two consecutive pages is printed using the starting page and the localization string <code>sequens</code> , e. g., the above list is compressed to ‘1 sq., 11–13, 21–24’. |
| <code>three+</code> | Similar in concept to <code>two+</code> but a sequence of exactly three consecutive pages is printed using the starting page and the localization string <code>sequentes</code> , e. g., the above list is compressed to ‘1 sq., 11 sqq., 21–24’. |
| <code>all+</code> | Similar in concept to <code>three+</code> but any sequence of consecutive pages is printed as an open-ended range, e. g., the above list is compressed to ‘1 sq., 11 sqq., 21 sqq.’. |

All styles support both Arabic and Roman numerals. In order to avoid potentially ambiguous lists, different sets of numerals will not be mixed when generating ranges, e. g., the list ‘iii, iv, v, 6, 7, 8’ is compressed to ‘iii–v, 6–8’.

`backrefsetstyle`=setonly, memonly, setormem, setandmem, memandset, setplusmem default: setonly

控制 `set` 条目如何反向连接。

This option controls how back references to `@set` entries and their members are handled. The following options are available:

- `setonly` All back references are added to the `@set` entry. The `pageref` lists of set members remain blank.
- `memonly` References to set members are added to the respective member. References to the `@set` entry are added to all members. The `pageref` list of the `@set` entry remains blank.
- `setormem` References to the `@set` entry are added to the `@set` entry. References to set members are added to the respective member.
- `setandmem` References to the `@set` entry are added to the `@set` entry. References to set members are added to the respective member and to the `@set` entry.
- `memandset` References to the `@set` entry are added to the `@set` entry and to all members. References to set members are added to the respective member.
- `setplusmem` References to the `@set` entry are added to the `@set` entry and to all members. References to set members are added to the respective member and to the `@set` entry.

`indexing`=true, false, cite, bib default: false

该选项控制在参考文献或引用中使用索引。

This option controls indexing in citations and in the bibliography. More precisely, it affects the `\ifciteindex` and `\ifbibindex` commands from § 4.6.2. The option is settable on a global, a per-type, or on a per-entry basis. The possible choices are:

- `true` Enable indexing globally.
- `false` Disable indexing globally.
- `cite` Enable indexing in citations only.
- `bib` Enable indexing in the bibliography only.

This feature requires support by the selected citation style. All standard styles which ship with this package support indexing of both citations and entries in the bibliography. Note that you still need to enable indexing globally with `\makeindex` to get an index.

`loadfiles=true, false` default: false

This option controls whether external files requested by way of the `\printfile` command are loaded. See also § 3.11.8 and `\printfile` in § 4.4.1. Note that this feature is disabled by default for performance reasons.

`refsection=none, part, chapter, section, subsection` default: none

该选项自动产生一个新的参考文献分部在文档划分为章节的时候。This option automatically starts a new reference section at a document division such as a chapter or a section. This is equivalent to the `\newrefsection` command, see § 3.6.5 for details. The following choice of document divisions is available:

- `none` Disable this feature.
- `part` Start a reference section at every `\part` command.
- `chapter` Start a reference section at every `\chapter` command.
- `section` Start a reference section at every `\section` command.
- `subsection` Start a reference section at every `\subsection` command.

带星号的文档划分章节命令不产生参考文献部分。

The starred versions of these commands will not start a new reference section.

`refsegment=none, part, chapter, section, subsection` default: none

类似于 `refsection` 选项。

Similar to the `refsection` option but starts a new reference segment. This is equivalent to the `\newrefsegment` command, see § 3.6.6 for details. When using both options, note that you can only apply this option to a lower-level document division than the one `refsection` is applied to and that nested reference segments will be local to the enclosing reference section.

`citereset=none, part, chapter, section, subsection` default: none

该选项控制在文档划分时自动执行 `\citereset` 命令。

This option automatically executes the `\citereset` command from § 3.7.8 at a document division such as a chapter or a section. The following choice of document divisions is available:

- `none` Disable this feature.
- `part` Perform a reset at every `\part` command.
- `chapter` Perform a reset at every `\chapter` command.
- `section` Perform a reset at every `\section` command.
- `subsection` Perform a reset at every `\subsection` command.

The starred versions of these commands will not trigger a reset.

`abbreviate=true, false`

default: `true`

是否在引用和参考文献中使用长或缩略的字符串。

Whether or not to use long or abbreviated strings in citations and in the bibliography. This option affects the localization modules. If this option is enabled, key terms such as ‘editor’ are abbreviated. If not, they are written out.

`date=year, short, long, terse, comp, iso8601`

default: `comp`

该选项控制打印日期的规范。

This option controls the basic format of printed date specifications. The following choices are available:

<code>year</code>	Use only years, for example: 2010 2010–2012
<code>short</code>	Use the short format with verbose ranges, for example: 01/01/2010 21/01/2010–30/01/2010 01/21/2010–01/30/2010
<code>long</code>	Use the long format with verbose ranges, for example: 1st January 2010 21st January 2010–30th January 2010 January 21, 2010–January 30, 2010
<code>terse</code>	Use the short format with compact ranges, for example: 21–30/01/2010 01/21–01/30/2010
<code>comp</code>	Use the long format with compact ranges, for example: 21st–30th January 2010 January 21–30, 2010
<code>iso8601</code>	Use extended ISO-8601 format (yyyy-mm-dd), for example: 2010-01-01 2010-01-21/2010-01-30

As seen in the above examples, the actual date format is language specific. Note that the month name in all long formats is responsive to the `abbreviate` package option. The leading zeros in all short formats may be controlled separately with the `datezeros` package option.

`datelabel=year, short, long, terse, comp, iso8601` default: year

类似域 `date` 选项，但控制由 `\DeclareLabeldate` 命令选择的日期域。

Similar to the `date` option but controls the format of the date field selected with `\DeclareLabeldate`.

`origdate=year, short, long, terse, comp, iso8601` default: comp

Similar to the `date` option but controls the format of the `origdate`.

`eventdate=year, short, long, terse, comp, iso8601` default: comp

Similar to the `date` option but controls the format of the `eventdate`.

`urldate=year, short, long, terse, comp, iso8601` default: short

Similar to the `date` option but controls the format of the `urldate`.

`alldates=year, short, long, terse, comp, iso8601`

设置上述所有的日期选项为相同值。Sets all of the above date options to the same value.

`datezeros=true, false` default: true

This option controls whether `short` and `terse` dates are printed with leading zeros.

`dateabbrev=true, false` default: true

This option controls whether `long` and `comp` dates are printed with long or abbreviated month names. The option is similar to the generic `abbreviate` option but specific to the date formatting.

`defernumbers=true, false` default: false

In contrast to standard LaTeX, the numeric labels generated by this package are normally assigned to the full list of references at the beginning of the document body. If this option is enabled, numeric labels (i.e., the `labelnumber` field discussed in § 4.2.4) are assigned the first time an entry is printed in any bibliography. See § 3.12.5 for further explanation. This option requires two LaTeX runs after the data has been exported to the `bb1` file by the backend (in addition to any other runs required by page breaks changing etc.). An important thing to note is that if you change the value of this option in your document (or the value of options which depend on this like some of the options to the `\printbibliography` macro, see § 3.6.2), then it is likely that you will need to delete your current `aux` file and re-run LaTeX to obtain the correct numbering. See § 4.1.

`punctfont=true, false` default: false

启用处理单位标点时的替换机制 This option enables an alternative mechanism for dealing with unit punctuation after a field printed in a different font (for example, a title printed in italics). See `\setpunctfont` in § 4.7.1 for details.

`arxiv=abs,ps,pdf,format`

default: `abs`

Path selector for arXiv links. If hyperlink support is enabled, this option controls which version of the document the arXiv eprint links will point to. The following choices are available:

- `abs` Link to the abstract page.
- `ps` Link to the PostScript version.
- `pdf` Link to the PDF version.
- `format` Link to the format selector page.

See § 3.11.7 for details on support for arXiv and electronic publishing information.

`texencoding=auto, <encoding>`

default: `auto`

指定 `tex` 文件的编码。

Specifies the encoding of the `tex` file. This option affects the data transferred from the backend to Biblatex. When using Biber, this corresponds to Biber's `--output_encoding` option. The following choices are available:

- `auto` Try to auto-detect the input encoding. If the `inputenc/inputenx/luainputenc` package is available, Biblatex will get the main encoding from that package. If not, it assumes UTF-8 encoding if XeTeX or LuaTeX has been detected, and Ascii otherwise.
- `<encoding>` Specifies the `<encoding>` explicitly. This is for odd cases in which auto-detection fails or you want to force a certain encoding for some reason.

Note that setting `texencoding=<encoding>` will also affect the `bibencoding` option if `bibencoding=auto`.

`bibencoding=auto, <encoding>`

default: `auto`

Specifies the encoding of the `bib` files. When using Biber, this corresponds to Biber's `--input_encoding` option. The following choices are available:

- `auto` Use this option if the workflow is transparent, i. e., if the encoding of the `bib` file is identical to the encoding of the `tex` file.
- `<encoding>` If the encoding of the `bib` file is different from the one of the `tex` file, you need to specify it explicitly.

By default, Biblatex assumes that the `tex` file and the `bib` file use the same encoding (`bibencoding=auto`). Note that some backends only support a limited number of encodings. See § 2.4.3 for further instructions.

`safeinputenc=true,false`

default: `false` Biber only

If this option is enabled, Biblatex will automatically force `texencoding=ascii` if the `inputenc/inputenx` package has been loaded and the input encoding is UTF-8,

i. e., it will ignore any macro-based UTF-8 support and use Ascii only. Biber will then try to convert any non-Ascii data in the `bib` file to Ascii. For example, it will convert `§` to `\d{S}`. See § 2.4.3.4 for an explanation of why you may want to enable this option.

`bibwarn=true, false` default: true

By default, Bibl_{at}ex will report warnings issued by the backend concerning the data in the `bib` file as LaTeX warnings. Use this option to suppress such warnings.

`mincrossrefs=<integer>` default: 2

Sets the minimum number of cross references to *<integer>* when requesting a backend run.¹⁵ Note that when using the BibTeX backend, this package option merely affects the format of certain requests written to the transcript file. It will not have any effect if the editor or compiler script launching BibTeX does not include dedicated Bibl_{at}ex support or if BibTeX is manually launched from the command-line.¹⁶ See § 2.4.4 for details. This option also affects the handling of the `xref` field. See the field description in § 2.2.3 as well as § 2.4.1 for details.

3.1.2.2 Style-specific The following options are provided by the standard styles (as opposed to the core package). Technically, they are preamble options like those in § 3.1.2.1.

下面的选项是标准样式提供的。技术上来说，它们是导言区选项。

`isbn=true, false` default: true

控制类似 `isbn` 的域是否打印。

This option controls whether the fields `isbn/issn/isrn` are printed.

`url=true, false` default: true

控制 `url` 域和访问日期是否打印。

This option controls whether the `url` field and the access date is printed. The option only affects entry types whose `url` information is optional. The `url` field of `@online` entries is always printed.

`doi=true, false` default: true

This option controls whether the field `doi` is printed.

控制 `doi` 域是否打印

`eprint=true, false` default: true

This option controls whether `eprint` information is printed.

控制 `eprint` 信息是否打印。

¹⁵If an entry which is cross-referenced by other entries in the `bib` file hits this threshold, it is included in the bibliography even if it has not been cited explicitly. This is a standard feature of the BibTeX (also Biber) and not specific to Bibl_{at}ex. See the description of the `crossref` field in § 2.2.3 for further information.

¹⁶As of this writing, no LaTeX editors or compiler scripts with dedicated Bibl_{at}ex support are known, but this will hopefully change in the future.

3.1.2.3 Internal The default settings of the following preamble options are controlled by bibliography and citation styles. Apart from the `pagetracker` and `firstinits` options, which you may want to adapt, there is normally no need to set them explicitly.

下面的导言区选项是由样式文件控制的。

`pagetracker=true, false, page, spread` default: false

This option controls the page tracker which is required by the `\ifsamepage` and `\iffirstonpage` tests from § 4.6.2. The possible choices are:

页码追踪器选项。

`true` Enable the tracker in automatic mode. This is like `spread` if LaTeX is in twoside mode, and like `page` otherwise.

`false` Disable the tracker.

`page` Enable the tracker in page mode. In this mode, tracking works on a per-page basis.

`spread` Enable the tracker in spread mode. In this mode, tracking works on a per-spread (double page) basis.

Note that this tracker is disabled in all floats, see § 4.11.5.

`citecounter=true, false, context` default: false

控制引用计数器的使用模式。

This option controls the citation counter which is required by `citecounter` from § 4.6.2. The possible choices are:

`true` Enable the citation counter in global mode.

`false` Disable the citation counter.

`context` Enable the citation counter in context-sensitive mode. In this mode, citations in footnotes and in the body text are counted independently.

`citetracker=true, false, context, strict, constrict` default: false

控制引用追踪器的使用模式。

This option controls the citation tracker which is required by the `\ifciteseen` and `\ifentryseen` tests from § 4.6.2. The possible choices are:

`true` Enable the tracker in global mode.

`false` Disable the tracker.

`context` Enable the tracker in context-sensitive mode. In this mode, citations in footnotes and in the body text are tracked independently.

`strict` Enable the tracker in strict mode. In this mode, an item is only considered by the tracker if it appeared in a stand-alone citation, i. e., if a single entry key was passed to the citation command.

`constrict` This mode combines the features of `context` and `strict`.

Note that this tracker is disabled in all floats, see § 4.11.5.

`ibidtracker=true, false, context, strict, constrict` default: false

控制出处追踪器的使用模式。

This option controls the ‘ibidem’ tracker which is required by the `\ifciteibid` test from § 4.6.2. The possible choices are:

`true` Enable the tracker in global mode.

`false` Disable the tracker.

`context` Enable the tracker in context-sensitive mode. In this mode, citations in footnotes and in the body text are tracked separately.

`strict` Enable the tracker in strict mode. In this mode, potentially ambiguous references are suppressed. A reference is considered ambiguous if either the current citation (the one including the ‘ibidem’) or the previous citation (the one the ‘ibidem’ refers to) consists of a list of references.¹⁷

`constrict` This mode combines the features of `context` and `strict`. It also keeps track of footnote numbers and detects potentially ambiguous references in footnotes in a stricter way than the `strict` option. In addition to the conditions imposed by the `strict` option, a reference in a footnote will only be considered as unambiguous if the current citation and the previous citation are given in the same footnote or in immediately consecutive footnotes.

Note that this tracker is disabled in all floats, see § 4.11.5.

`opcitracker=true, false, context, strict, constrict` default: false

前述引用追踪器的使用模式。

This option controls the ‘opcit’ tracker which is required by the `\ifopcit` test from § 4.6.2. This feature is similar to the ‘ibidem’ tracker, except that it tracks citations on a per-author/editor basis, i. e., `\ifopcit` will yield `true` if the cited item is the same as the last one by this author/editor. The possible choices are:

`true` Enable the tracker in global mode.

`false` Disable the tracker.

¹⁷For example, suppose the initial citation is “Jones, *Title*; Williams, *Title*” and the following one “ibidem”. From a technical point of view, it is fairly clear that the ‘ibidem’ refers to ‘Williams’ because this is the last reference processed by the previous citation command. To a human reader, however, this may not be obvious because the ‘ibidem’ may also refer to both titles. The strict mode avoids such ambiguous references.

- `context` Enable the tracker in context-sensitive mode. In this mode, citations in footnotes and in the body text are tracked separately.
- `strict` Enable the tracker in strict mode. In this mode, potentially ambiguous references are suppressed. See `ibidtracker=strict` for details.
- `constrict` This mode combines the features of `context` and `strict`. See the explanation of `ibidtracker=constrict` for details.

Note that this tracker is disabled in all floats, see § 4.11.5.

`loccittracker=true, false, context, strict, constrict` default: false

引用位置追踪器。

This option controls the ‘loccit’ tracker which is required by the `\ifloccit` test from § 4.6.2. This feature is similar to the ‘opcit’ tracker except that it also checks whether the $\langle postnote \rangle$ arguments match, i.e., `\ifloccit` will yield `true` if the citation refers to the same page cited before. The possible choices are:

- `true` Enable the tracker in global mode.
- `false` Disable the tracker.
- `context` Enable the tracker in context-sensitive mode. In this mode, citations in footnotes and in the body text are tracked separately.
- `strict` Enable the tracker in strict mode. In this mode, potentially ambiguous references are suppressed. See `ibidtracker=strict` for details. In addition to that, this mode also checks if the $\langle postnote \rangle$ argument is numerical (based on `\ifnumerals` from § 4.6.2).
- `constrict` This mode combines the features of `context` and `strict`. See the explanation of `ibidtracker=constrict` for details. In addition to that, this mode also checks if the $\langle postnote \rangle$ argument is numerical (based on `\ifnumerals` from § 4.6.2).

Note that this tracker is disabled in all floats, see § 4.11.5.

`idemtracker=true, false, context, strict, constrict` default: false

同前追踪器。

This option controls the ‘idem’ tracker which is required by the `\ifciteidem` test from § 4.6.2. The possible choices are:

- `true` Enable the tracker in global mode.
- `false` Disable the tracker.
- `context` Enable the tracker in context-sensitive mode. In this mode, citations in footnotes and in the body text are tracked separately.

`strict` This is an alias for `true`, provided only for consistency with the other trackers. Since ‘idem’ replacements do not get ambiguous in the same way as ‘ibidem’ or ‘op. cit.’, the `strict` tracking mode does not apply to them.

`constrict` This mode is similar to `context` with one additional condition: a reference in a footnote will only be considered as unambiguous if the current citation and the previous citation are given in the same footnote or in immediately consecutive footnotes.

Note that this tracker is disabled in all floats, see § 4.11.5.

`parentracker=true, false` default: true

圆括号追踪器。

This option controls the parenthesis tracker which keeps track of nested parentheses and brackets. This information is used by `\parentext` and `\brackettext` from § 3.7.5, `\mkbibparens` and `\mkbibbrackets` from § 4.10.4 and `\bibopenparen`, `\bibcloseparen`, `\bibopenbracket`, `\bibclosebracket` (also § 4.10.4).

`maxparens=<integer>` default: 3

括号嵌套的最大层级。

The maximum permitted nesting level of parentheses and brackets. If parentheses and brackets are nested deeper than this value, Biblatex will issue errors.

`firstinits=true, false` default: false

启用名字由首字母表示。

When enabled, all first and middle names will be rendered as initials. The option will affect the `\iffirstinits` test from § 4.6.2.

`sortfirstinits=true, false` default: false

姓名排序时，利用首字母进行。

When enabled, sorting names will only use their initials. This is separate from `firstinits` in case users want to show only initials but sort on full names, for example. Biber only

`terseinits=true, false` default: false

控制 biblatex 生成的首字母的格式。biber 后端可以重定义一些宏来控制格式，详见 § 3.12.4。

This option controls the format of initials generated by Biblatex. If enabled, initials are rendered using a terse format without dots and spaces. For example, the initials of Donald Ervin Knuth would be rendered as ‘D. E.’ by default, and as ‘DE’ if this option

is enabled. The option will affect the `\iffterseinit`s test from § 4.6.2. With Biber, the option works by redefining some macros which control the format of initials. See § 3.12.4 for details. Biber only

`labelalpha=true, false` default: false

是否提供特殊的域，详见 § 4.2.4。

Whether or not to provide the special fields `labelalpha` and `extraalpha`, see § 4.2.4 for details. With Biber, this option is also settable on a per-type basis. See also `maxalphanames` and `minalphanames`. Table 5 summarises the various `extra*` disambiguation counters and what they track. Biber only

`maxalphanames=<integer>` default: 3 Biber only

Similar to the `maxnames` option but customizes the format of the `labelalpha` field.

`minalphanames=<integer>` default: 1 Biber only

Similar to the `minnames` option but customizes the format of the `labelalpha` field.

`labelnumber=true, false` default: false

是否提供特殊域`labelnumber`。Whether or not to provide the special field `labelnumber`, see § 4.2.4 for details. This option is also settable on a per-type basis.

`labeltitle=true, false` default: false Biber only

是否提供特殊域`extratitle`。Whether or not to provide the special field `extratitle`, see § 4.2.4 for details. Note that the special field `labeltitle` is always provided and this option controls rather whether `labeltitle` is used to generate `extratitle` information. This option is also settable on a per-type basis. Table 5 summarises the various `extra*` disambiguation counters and what they track.

`labeltitleyear=true, false` default: false Biber only

Whether or not to provide the special field `extratitleyear`, see § 4.2.4 for details. Note that the special field `labeltitle` is always provided and this option controls rather whether `labeltitle` is used to generate `extratitleyear` information. This option is also settable on a per-type basis. Table 5 summarises the various `extra*` disambiguation counters and what they track.

`labeldate=true, false` default: false

Whether or not to provide the special fields `labelyear`, `labelmonth`, `labelday` and `extrayear`, see § 4.2.4 for details. With Biber, this option is also settable on a per-type basis. Table 5 summarises the various `extra*` disambiguation counters and what they track. Biber only

`singletitle=true, false`

default: false

Whether or not to provide the data required by the `\ifsingletitle` test, see § 4.6.2 for details. With Biber, this option is also settable on a per-type basis.

Biber only

`uniquename=true, false, init, full, allinit, allfull, mininit, minfull`

default: false

Biber only

是否更新uniquename计数器。

Whether or not to update the `uniquename` counter, see § 4.6.2 for details. This feature will disambiguate individual names in the `labelname` list. This option is also settable on a per-type basis. The possible choices are:

- `true` An alias for `full`.
- `false` Disable this feature.
- `init` Disambiguate names using initials only.
- `full` Disambiguate names using initials or full names, as required.
- `allinit` Similar to `init` but disambiguates all names in the `labelname` list, beyond `maxnames/minnames/uniquelist`.
- `allfull` Similar to `full` but disambiguates all names in the `labelname` list, beyond `maxnames/minnames/uniquelist`.
- `mininit` A variant of `init` which only disambiguates names in lists with identical last names.
- `minfull` A variant of `full` which only disambiguates names in lists with identical last names.

Note that the `uniquename` option will also affect `uniquelist`, the `\ifsingletitle` test, and the `extrayear` field. See § 4.11.4 for further details and practical examples.

`uniquelist=true, false, minyear`

default: false

Biber only

是否更新uniquelist计数器。

Whether or not to update the `uniquelist` counter, see § 4.6.2 for details. This feature will disambiguate the `labelname` list if it has become ambiguous after `maxnames/minnames` truncation. Essentially, it overrides `maxnames/minnames` on a per-field basis. This option is also settable on a per-type basis. The possible choices are:

- `true` Disambiguate the `labelname` list.
- `false` Disable this feature.
- `minyear` Disambiguate the `labelname` list only if the truncated list is identical to another one with the same `labelyear`. This mode of operation is useful for author-year styles and requires `labeldate=true`.

Option	Enabled field	Enabled counter	Counter tracks
labelalpha	labelalpha	extraalpha	label
labeldate	labelyear	extrayear	labelname+labelyear
labeltitle	–	extratitle	labelname+labeltitle
labeltitleyear	–	extratitleyear	labeltitle+labelyear

Table 5: Disambiguation counters

Note that the `uniquelist` option will also affect the `\ifsingletitle` test and the `extrayear` field. See § 4.11.4 for further details and practical examples.

3.1.3 条目选项 Entry Options

Entry options are package options which determine how bibliography data entries are handled. They may be set at various scopes defined below.

条目选项是控制参考文献数据条目处理的包选项。

3.1.3.1 Preamble/Type/Entry Options The following options are settable on a per-type basis or on a per-entry in the `options` field. In addition to that, they may also be used in the optional argument to `\usepackage` as well as in the configuration file and the document preamble. This is useful if you want to change the default behaviour globally.

下面的选项是可以根据条目和类型重设的。而且也可以在 `\usepackage` 命令中作为选项使用。如果需要改变默认的全局行为会非常有用。

`useauthor=true, false` default: true

是否在标签中使用 `author` 或者在排序中考虑。

Whether the `author` is used in labels and considered during sorting. This may be useful if an entry includes an `author` field but is usually not cited by author for some reason. Setting `useauthor=false` does not mean that the `author` is ignored completely. It means that the `author` is not used in labels and ignored during sorting. The entry will then be alphabetized by `editor` or `title`. With the standard styles, the `author` is printed after the title in this case. See also § 3.5. With Biber, this option is also settable on a per-type and per-entry basis.

Biber only

`useeditor=true, false` default: true

是否用 `editor` 代替缺失的 `author`。

Whether the `editor` replaces a missing `author` in labels and during sorting. This may be useful if an entry includes an `editor` field but is usually not cited by editor. Setting `useeditor=false` does not mean that the `editor` is ignored completely. It means that the `editor` does not replace a missing `author` in labels and during sorting. The entry will then be alphabetized by `title`. With the standard styles, the `editor` is printed after the title in this case. See also § 3.5. With Biber, this option is

Biber only

also settable on a per-type and per-entry basis.

`usetranslator=true, false`

default: false

Whether the translator replaces a missing author/editor in labels and during sorting. Setting `usetranslator=true` does not mean that the translator overrides the author/editor. It means that the translator is considered as a fallback if the author/editor is missing or if `useauthor` and `useeditor` are set to false. In other words, in order to cite a book by translator rather than by author, you need to set the following options: With Biber, this option is also settable on a per-type and per-entry basis.

Biber only

```
@Book{...,
  options      = {useauthor=false,usetranslator=true},
  author       = {...},
  translator    = {...},
  ...
```

With the standard styles, the translator is printed after the title by default. See also § 3.5.

`use<name>=true, false`

default: true

As per `useauthor`, `useeditor` and `usetranslator`, all name lists defined in the data model have an option controlling their behaviour in sorting and labelling automatically defined. Global, per-type and per-entry options called ‘`use<name>`’ are automatically created.

`useprefix=true, false`

default: false

Whether the name prefix (von, van, of, da, de, della, etc.) is considered when printing the last name in citations. This also affects the sorting and formatting of the bibliography as well as the generation of certain types of labels. If this option is enabled, BibLaTeX always precedes the last name with the prefix. For example, Ludwig van Beethoven would be cited as “Beethoven” and alphabetized as “Beethoven, Ludwig van” by default. If this option is enabled, he is cited as “van Beethoven” and alphabetized as “Van Beethoven, Ludwig” instead. With Biber, this option is also settable on a per-type basis.

Biber only

`indexing=true, false, cite, bib`

The `indexing` option is also settable per-type or per-entry basis. See § 3.1.2.1 for details.

3.1.3.2 Type/Entry Options The following options are settable on a per-type basis or on a per-entry in the `options` field. They are not available globally.

`skipbib=true, false` default: false

If this option is enabled, the entry is excluded from the bibliography but it may still be cited. With Biber, this option is also settable on a per-type basis.

Biber only

`skiplos=true, false` default: false

BibTeX only

If this option is enabled, the entry is excluded from the list of shorthands. It is still included in the bibliography and it may also be cited by shorthand. This option is deprecated when using Biber as the backend. Use `skipbiblist` instead.

`skipbiblist=true, false` default: false

Biber only

This is the same as the `skiplos` option when using the BibTeX backend. It is renamed for Biber to be consistent with the more generalised bibliography list functionality See § 3.6.4. If this option is enabled, the entry is excluded from the list of shorthands. It is still included in the bibliography and it may also be cited by shorthand. This option is also settable on a per-type basis.

`skiplab=true, false` default: false

If this option is enabled, Biblatex will not assign any labels to the entry. It is not required for normal operation. Use it with care. If enabled, Biblatex can not guarantee unique citations for the respective entry and citations styles which require labels may fail to create valid citations for the entry. With Biber, this option is also settable on a per-type basis.

Biber only

`dataonly=true, false` default: false

Setting this option is equivalent to `uniquename=false, uniquelist=false, skipbib, skiplos/skipbiblist, and skiplab`. It is not required for normal operation. Use it with care. With Biber, this option is also settable on a per-type basis.

Biber only

3.1.3.3 Entry Only Options The following options are settable only on a per-entry in the `options` field. They are not available globally or per-type.

`labelnamefield=<fieldname>`

Specifies the field to consider first when looking for a `labelname` candidate. It is essentially prepended to the search list created by `\DeclareLabelname` for just this entry.

`labeltitlefield=<fieldname>`

Specifies the field to consider first when looking for a `labeltitle` candidate. It is essentially prepended to the search list created by `\DeclareLabeltitle` for just this entry.

3.1.4 遗留选项 Legacy Options

下面的遗留选项可以在`\documentclass`中全局使用或者在`\usepackage`中作为可选参数。

`openbib` 这一选项用于兼容标准的 LaTeX 文档类。`openbib`类似于 `block=par`。

Deprecated

3.2 全局定制 Global Customization

除了写新的标注和参考文献样式，还有多种方式来定制本宏包提供的样式。定制可以在导言区中进行，也可以在配置文件中进行以便长期使用。配置文件也用来初始化与默认值不同的包选项。

3.2.1 配置文件 Configuration File

如果提供该文件，则包将加载该配置文件`biblatex.cfg`。该文件在包末尾读入，紧跟在标注和参考文献样式后面。

3.2.2 设置包选项 Setting Package Options

§ 3.1.1节的加载用包选项必须在`\usepackage`的可选参数中给出。§ 3.1.2给出的包选项也需要在导言区给出。这些选项与如下命令一起使用：

```
\ExecuteBibliographyOptions[⟨entrytype, ...⟩]{⟨key=value, ...⟩}
```

This command may also be used in the configuration file to modify the default setting of a package option. Certain options are also settable on a per-type basis. In this case, the optional `⟨entrytype⟩` argument specifies the entry type. The `⟨entrytype⟩` argument may be a comma-separated list of values.

3.3 标准样式 Standard Styles

This section provides a short description of all bibliography and citation styles which ship with the Biblatex package. If you want to write your own styles, see § 4.

3.3.1 标注样式 Citation Styles

The citation styles which come with this package implement several common citation schemes. All standard styles cater for the `shorthand` field and support hyperlinks as well as indexing.

`numeric` This style implements a numeric citation scheme similar to the standard bibliographic facilities of LaTeX. It should be employed in conjunction with a numeric bibliography style which prints the corresponding labels in the bibliography. It is intended for in-text citations. The style will set the following package options at load time: `autocite=inline`, `labelnumber=true`. This style also provides an additional preamble option called `subentry` which affects the handling of entry sets. If this option is disabled, citations referring to a member of a set will point to the entire set. If it is enabled,

the style supports citations like “[5c]” which point to a subentry in a set (the third one in this example). See the style example for details.

- numeric-comp** A compact variant of the `numeric` style which prints a list of more than two consecutive numbers as a range. This style is similar to the `cite` package and the `sort&compress` option of the `natbib` package in numerical mode. For example, instead of “[8, 3, 1, 7, 2]” this style would print “[1–3, 7, 8]”. It is intended for in-text citations. The style will set the following package options at load time: `autocite=inline`, `sortcites=true`, `labelnumber=true`. It also provides the `subentry` option.
- numeric-verb** A verbose variant of the `numeric` style. The difference affects the handling of a list of citations and is only apparent when multiple entry keys are passed to a single citation command. For example, instead of “[2, 5, 6]” this style would print “[2]; [5]; [6]”. It is intended for in-text citations. The style will set the following package options at load time: `autocite=inline`, `labelnumber=true`. It also provides the `subentry` option.
- alphabetic** This style implements an alphabetic citation scheme similar to the `alpha.bst` style of traditional BibTeX. The alphabetic labels resemble a compact author-year style to some extent, but the way they are employed is similar to a numeric citation scheme. For example, instead of “Jones 1995” this style would use the label “[Jon95]”. “Jones and Williams 1986” would be rendered as “[JW86]”. This style should be employed in conjunction with an alphabetic bibliography style which prints the corresponding labels in the bibliography. It is intended for in-text citations. The style will set the following package options at load time: `autocite=inline`, `labelalpha=true`.
- alphabetic-verb** A verbose variant of the `alphabetic` style. The difference affects the handling of a list of citations and is only apparent when multiple entry keys are passed to a single citation command. For example, instead of “[Doe92; Doe95; Jon98]” this style would print “[Doe92]; [Doe95]; [Jon98]”. It is intended for in-text citations. The style will set the following package options at load time: `autocite=inline`, `labelalpha=true`.
- authoryear** This style implements an author-year citation scheme. If the bibliography contains two or more works by the same author which were all published in the same year, a letter is appended to the year. For example, this style would print citations such as “Doe 1995a; Doe 1995b; Jones 1998”. This style should be employed in conjunction with an author-year bibliography style which prints the corresponding labels in the bibliography. It is primarily intended for in-text citations, but it could also be used with citations given in footnotes. The style will set the following package options at load time: `autocite=inline`, `labeldate=true`, `uniquename=full`, `uniquelist=true`.
- authoryear-comp** A compact variant of the `authoryear` style which prints the author only once if subsequent references passed to a single citation command share the same author. If they

share the same year as well, the year is also printed only once. For example, instead of “Doe 1995b; Doe 1992; Jones 1998; Doe 1995a” this style would print “Doe 1992, 1995a,b; Jones 1998”. It is primarily intended for in-text citations, but it could also be used with citations given in footnotes. The style will set the following package options at load time: `autocite=inline`, `sortcites=true`, `labeldate=true`, `uniquename=full`, `uniquelist=true`.

authoryear-ibid A variant of the `authoryear` style which replaces repeated citations by the abbreviation *ibidem* unless the citation is the first one on the current page or double-page spread, or the *ibidem* would be ambiguous in the sense of the package option `ibidtracker=constrict`. The style will set the following package options at load time: `autocite=inline`, `labeldate=true`, `uniquename=full`, `uniquelist=true`, `ibidtracker=constrict`, `pagetracker=true`. This style also provides an additional preamble option called `ibidpage`. See the style example for details.

authoryear-icomp A style combining `authoryear-comp` and `authoryear-ibid`. The style will set the following package options at load time: `autocite=inline`, `labeldate=true`, `uniquename=full`, `uniquelist=true`, `ibidtracker=constrict`, `pagetracker=true`, `sortcites=true`. This style also provides an additional preamble option called `ibidpage`. See the style example for details.

authortitle This style implements a simple author-title citation scheme. It will make use of the `shorttitle` field, if available. It is intended for citations given in footnotes. The style will set the following package options at load time: `autocite=footnote`, `uniquename=full`, `uniquelist=true`.

authortitle-comp A compact variant of the `authortitle` style which prints the author only once if subsequent references passed to a single citation command share the same author. For example, instead of “Doe, *First title*; Doe, *Second title*” this style would print “Doe, *First title*, *Second title*”. It is intended for citations given in footnotes. The style will set the following package options at load time: `autocite=footnote`, `sortcites=true`, `uniquename=full`, `uniquelist=true`.

authortitle-ibid A variant of the `authortitle` style which replaces repeated citations by the abbreviation *ibidem* unless the citation is the first one on the current page or double-page spread, or the *ibidem* would be ambiguous in the sense of the package option `ibidtracker=constrict`. It is intended for citations given in footnotes. The style will set the following package options at load time: `autocite=footnote`, `uniquename=full`, `uniquelist=true`, `ibidtracker=constrict`, `pagetracker=true`. This style also provides an additional preamble option called `ibidpage`. See the style example for details.

authortitle-icomp A style combining the features of `authortitle-comp` and `authortitle-ibid`. The style will set the following package options at load time: `autocite=footnote`,

`uniquename = full, uniquelist = true, ibidtracker = constrict, pagetracker=true, sortcites=true`. This style also provides an additional preamble option called `ibidpage`. See the style example for details.

authortitle-terse A terse variant of the `authortitle` style which only prints the title if the bibliography contains more than one work by the respective author/editor. This style will make use of the `shorttitle` field, if available. It is suitable for in-text citations as well as citations given in footnotes. The style will set the following package options at load time: `autocite=inline, singletitle=true, uniquename=full, uniquelist=true`.

authortitle-tcomp A style combining the features of `authortitle-comp` and `authortitle-terse`. This style will make use of the `shorttitle` field, if available. It is suitable for in-text citations as well as citations given in footnotes. The style will set the following package options at load time: `autocite=inline, sortcites=true, singletitle=true, uniquename=full, uniquelist=true`.

authortitle-ticomp A style combining the features of `authortitle-icomp` and `authortitle-terse`. In other words: a variant of the `authortitle-tcomp` style with an *ibidem* feature. This style is suitable for in-text citations as well as citations given in footnotes. It will set the following package options at load time: `autocite = inline, ibidtracker = constrict, pagetracker = true, sortcites=true, singletitle=true, uniquename=full, uniquelist=true`. This style also provides an additional preamble option called `ibidpage`. See the style example for details.

verbose A verbose citation style which prints a full citation similar to a bibliography entry when an entry is cited for the first time, and a short citation afterwards. If available, the `shorttitle` field is used in all short citations. If the `shorthand` field is defined, the shorthand is introduced on the first citation and used as the short citation thereafter. This style may be used without a list of references and shorthands since all bibliographic data is provided on the first citation. It is intended for citations given in footnotes. The style will set the following package options at load time: `autocite=footnote, citetracker=context`. This style also provides an additional preamble option called `citepages`. See the style example for details.

verbose-ibid A variant of the `verbose` style which replaces repeated citations by the abbreviation *ibidem* unless the citation is the first one on the current page or double-page spread, or the *ibidem* would be ambiguous in the sense of `ibidtracker=strict`. This style is intended for citations given in footnotes. The style will set the following package options at load time: `autocite=footnote, citetracker=context, ibidtracker=constrict, pagetracker=true`. This style also provides additional preamble options called `ibidpage` and `citepages`. See the style example for details.

- verbose-note** This style is similar to the `verbose` style in that it prints a full citation similar to a bibliography entry when an entry is cited for the first time, and a short citation afterwards. In contrast to the `verbose` style, the short citation is a pointer to the footnote with the full citation. If the bibliography contains more than one work by the respective author/editor, the pointer also includes the title. If available, the `shorttitle` field is used in all short citations. If the `shorthand` field is defined, it is handled as with the `verbose` style. This style may be used without a list of references and shorthands since all bibliographic data is provided on the first citation. It is exclusively intended for citations given in footnotes. The style will set the following package options at load time: `autocite=footnote`, `citetracker=context`, `singletitle=true`. This style also provides additional preamble options called `pageref` and `citepages`. See the style example for details.
- verbose-inote** A variant of the `verbose-note` style which replaces repeated citations by the abbreviation *ibidem* unless the citation is the first one on the current page or double-page spread, or the *ibidem* would be ambiguous in the sense of `ibidtracker=strict`. This style is exclusively intended for citations given in footnotes. It will set the following package options at load time: `autocite=footnote`, `citetracker=context`, `ibidtracker=constrict`, `singletitle=true`, `pagetracker=true`. This style also provides additional preamble options called `ibidpage`, `pageref`, and `citepages`. See the style example for details.
- verbose-trad1** This style implements a traditional citation scheme. It is similar to the `verbose` style in that it prints a full citation similar to a bibliography entry when an item is cited for the first time, and a short citation afterwards. Apart from that, it uses the scholarly abbreviations *ibidem*, *idem*, *op. cit.*, and *loc. cit.* to replace recurrent authors, titles, and page numbers in repeated citations in a special way. If the `shorthand` field is defined, the shorthand is introduced on the first citation and used as the short citation thereafter. This style may be used without a list of references and shorthands since all bibliographic data is provided on the first citation. It is intended for citations given in footnotes. The style will set the following package options at load time: `autocite=footnote`, `citetracker=context`, `ibidtracker=constrict`, `idemtracker=constrict`, `opcitracker=context`, `loccitracker=context`. This style also provides additional preamble options called `ibidpage`, `strict`, and `citepages`. See the style example for details.
- verbose-trad2** Another traditional citation scheme. It is also similar to the `verbose` style but uses scholarly abbreviations like *ibidem* and *idem* in repeated citations. In contrast to the `verbose-trad1` style, the logic of the *op. cit.* abbreviations is different in this style and *loc. cit.* is not used at all. It is in fact more similar to `verbose-ibid` and `verbose-inote` than to `verbose-trad1`. The style will set the following package options at load time: `autocite=footnote`, `citetracker=context`, `ibidtracker=constrict`, `idemtracker=constrict`. This style also pro-

vides additional preamble options called `ibidpage`, `strict`, and `citepages`. See the style example for details.

verbose-trad3 Yet another traditional citation scheme. It is similar to the `verbose-trad2` style but uses the scholarly abbreviations *ibidem* and *op. cit.* in a slightly different way. The style will set the following package options at load time: `autocite=footnote`, `citetracker=context`, `ibidtracker=constrict`, `loccitracker=constrict`. This style also provides additional preamble options called `strict` and `citepages`. See the style example for details.

reading A citation style which goes with the bibliography style by the same name. It simply loads the `authortitle` style.

The following citation styles are special purpose styles. They are not intended for the final version of a document:

draft A draft style which uses the entry keys in citations. The style will set the following package options at load time: `autocite=plain`.

debug This style prints the entry key rather than some kind of label. It is intended for debugging only and will set the following package options at load time: `autocite=plain`.

3.3.2 参考文献样式 Bibliography Styles

All bibliography styles which come with this package use the same basic format for the individual bibliography entries. They only differ in the kind of label printed in the bibliography and the overall formatting of the list of references. There is a matching bibliography style for every citation style. Note that some bibliography styles are not mentioned below because they simply load a more generic style. For example, the bibliography style `authortitle-comp` will load the `authortitle` style.

numeric This style prints a numeric label similar to the standard bibliographic facilities of LaTeX. It is intended for use in conjunction with a numeric citation style. Note that the `shorthand` field overrides the default label. The style will set the following package options at load time: `labelnumber=true`. This style also provides an additional preamble option called `subentry` which affects the formatting of entry sets. If this option is enabled, all members of a set are marked with a letter which may be used in citations referring to a set member rather than the entire set. See the style example for details.

alphabetic This style prints an alphabetic label similar to the `alpha.bst` style of traditional BibTeX. It is intended for use in conjunction with an alphabetic citation style. Note that the `shorthand` field overrides the default label. The style will set the following package options at load time: `labelalpha=true`, `sorting=anyt`.

- authoryear** This style differs from the other styles in that the publication date is not printed towards the end of the entry but rather after the author/editor. It is intended for use in conjunction with an author-year citation style. Recurring author and editor names are replaced by a dash unless the entry is the first one on the current page or double-page spread. This style provides an additional preamble option called `dashed` which controls this feature. It also provided a preamble option called `mergedate`. See the style example for details. The style will set the following package options at load time: `labeldate=true, sorting=nyt, pagetracker=true, mergedate=true`.
- authortitle** This style does not print any label at all. It is intended for use in conjunction with an author-title citation style. Recurring author and editor names are replaced by a dash unless the entry is the first one on the current page or double-page spread. This style also provides an additional preamble option called `dashed` which controls this feature. See the style example for details. The style will set the following package options at load time: `pagetracker=true`.
- verbose** This style is similar to the `authortitle` style. It also provides an additional preamble option called `dashed`. See the style example for details. The style will set the following package options at load time: `pagetracker=true`.
- reading** This special bibliography style is designed for personal reading lists, annotated bibliographies, and similar applications. It optionally includes the fields `annotation`, `abstract`, `library`, and `file` in the bibliography. If desired, it also adds various kinds of short headers to the bibliography. This style also provides the additional preamble options `entryhead`, `entrykey`, `annotation`, `abstract`, `library`, and `file` which control whether or not the corresponding items are printed in the bibliography. See the style example for details. See also § 3.11.8. The style will set the following package options at load time: `loadfiles=true, entryhead=true, entrykey=true, annotation=true, abstract=true, library=true, file=true`.

The following bibliography styles are special purpose styles. They are not intended for the final version of a document:

- draft** This draft style includes the entry keys in the bibliography. The bibliography will be sorted by entry key. The style will set the following package options at load time: `sorting=debug`.
- debug** This style prints all bibliographic data in tabular format. It is intended for debugging only and will set the following package options at load time: `sorting=debug`.

3.4 相关条目 Related Entries

Almost all bibliography styles require authors to specify certain types of relationship between entries such as “Reprint of”, “Reprinted in” etc. It is impossible to provide data fields to cover all of these relationships and so Biblatex provides a general mechanism

for this using the entry fields `related`, `relatedtype` and `relatedstring`. A related entry does not need to be cited and does not appear in the bibliography itself (unless of course it is also cited itself independently) as a clone is taken of the related entry to be used as a data source. The `relatedtype` field should specify a localization string which will be printed before the information from the related entries is printed, for example “Orig. Pub. as”. The `relatedstring` field can be used to override the string determined via `relatedtype`. Some examples:

```
@Book{key1,
  ...
  related      = {key2},
  relatedtype = {reprintof},
  ...
}
```

```
@Book{key2,
  ...
}
```

Here we specify that entry `key1` is a reprint of entry `key2`. In the bibliography driver for Book entries, when `\usebibmacro{related}` is called for entry `key1`:

- If the localization string “reprintof” is defined, it is printed in the `relatedstring:reprintof` format. If this formatting directive is undefined, the string is printed in the `relatedstring:default` format.
- If the `related:reprintof` macro is defined, it is used to format the information contained in entry `key2`, otherwise the `related:default` macro is used
- If the `related:reprintof` format is defined, it is used to format both the localization string and data. If this format is not defined, then the `related` format is used instead.

It is also supported to have cascading and/or circular relations:

```
@Book{key1,
  ...
  related      = {key2},
  relatedtype = {reprintof},
  ...
}
```

```
@Book{key2,
  ...
}
```

```

    related      = {key3},
    relatedtype = {translationof},
    ...
}

@Book{key3,
    ...
    related      = {key2},
    relatedtype = {translatedas},
    ...
}

```

Multiple relations to the same entry are also possible:

```

@MVBook{key1,
    ...
    related      = {key2, key3},
    relatedtype = {multivolume},
    ...
}

@Book{key2,
    ...
}

@Book{key3,
    ...
}

```

Note the the order of the keys in lists of multiple related entries is important. The data from multiple related entries is printed in the order of the keys listed in this field. See § 4.5.1 for a more details on the mechanisms behind this feature. You can turn this feature off using the package option `related` from § 3.1.2.1.

You can use the `relatedoptions` to set options on the related entry data clone. This is useful if you need to override the `dataonly` option which is set by default on all related entry clones. For example, if you will expose some of the names in the related clone in your document, you may want to have them disambiguated from names in other entries but normally this won't happen as related clones have the per-entry `dataonly` option set and this in turn sets `uniquename=false` and `uniquelist=false`. In such a case, you can set `relatedoptions` to just `skiplab`, `skipbib`, `skiplos/skipbiblist`.

3.5 排序选项 Sorting Options

This package supports various sorting schemes for the bibliography. The sorting scheme is selected with the `sorting` package option from § 3.1.2.1. Apart from the regular data fields there are also some special fields which may be used to optimize the sorting of the bibliography. Appendices C.1 and C.2 give an outline of the alphabetic sorting schemes supported by Biblatex. Chronological sorting schemes are listed in appendix C.3. A few explanations concerning these schemes are in order.

The first item considered in the sorting process is always the `presort` field of the entry. If this field is undefined, Biblatex will use the default value ‘mm’ as a `presort` string. The next item considered is the `sortkey` field. If this field is defined, it serves as the master sort key. Apart from the `presort` field, no further data is considered in this case. If the `sortkey` field is undefined, sorting continues with the name. The package will try using the `sortname`, `author`, `editor`, and `translator` fields, in this order. Which fields are considered also depends on the setting of the `use<name>` options. If all such options are disabled, the `sortname` field is ignored as well. Note that all name fields are responsive to `maxnames` and `minnames`. If no name field is available, either because all of them are undefined or because all `use<name>` options are disabled, Biblatex will fall back to the `sorttitle` and `title` fields as a last resort. The remaining items are, in various order: the `sortyear` field, if defined, or the first four digits of the `year` field otherwise; the `sorttitle` field, if defined, or the `title` field otherwise; the `volume` field, which is padded to four digits with leading zeros, or the string 0000 otherwise. Note that the sorting schemes shown in appendix C.2 include an additional item: `labelalpha` is the label used by ‘alphabetic’ bibliography styles. Strictly speaking, the string used for sorting is `labelalpha + extraalpha`. The sorting schemes in appendix C.2 are intended to be used in conjunction with alphabetic styles only.

The chronological sorting schemes presented in appendix C.3 also make use of the `presort` and `sortkey` fields, if defined. The next item considered is the `sortyear` or the `year` field, depending on availability. The `ynt` scheme extracts the first four Arabic figures from the field. If both fields are undefined, the string 9999 is used as a fallback value. This means that all entries without a year will be moved to the end of the list. The `ydnt` scheme is similar in concept but sorts the year in descending order. As with the `ynt` scheme, the string 9999 is used as a fallback value. The remaining items are similar to the alphabetic sorting schemes discussed above. Note that the `ydnt` sorting scheme will only sort the date in descending order. All other items are sorted in ascending order as usual.

Using special fields such as `sortkey`, `sortname`, or `sorttitle` is usually not required. The Biblatex package is quite capable of working out the desired sorting order by using the data found in the regular fields of an entry. You will only need them if you want to manually modify the sorting order of the bibliography or if any data required for sorting is missing. Please refer to the field descriptions in § 2.2.3 for details on

possible uses of the special fields. Also note that using Biber instead of legacy BibTeX is strongly recommended.

3.6 参考文献命令 Bibliography Commands

3.6.1 文献数据源 Resources

`\addbibresource` [*options*] {*resource*}

Adds a *resource*, such as a `.bib` file, to the default resource list. This command is only available in the preamble. It replaces the `\bibliography` legacy command. Note that files must be specified with their full name, including the extension. Do not omit the `.bib` extension from the filename. Also note that the *resource* is a single resource. Invoke `\addbibresource` multiple times to add more resources, for example:

```
\addbibresource{bibfile1.bib}
\addbibresource{bibfile2.bib}
\addbibresource[location=remote]{http://www.citeulike.org/bibtex/group/9517}
\addbibresource[location=remote,label=lan]{ftp://192.168.1.57/~user/file.bib}
```

Since the *resource* string is read in a verbatim-like mode, it may contain arbitrary characters. The only restriction is that any curly braces must be balanced. The following *options* are available:

`label`=*identifier*

Assigns a label to a resource. The *identifier* may be used in place of the full resource name in the optional argument of `refsection` (see § 3.6.5).

`location`=*location* default: local

The location of the resource. The *location* may be either `local` for local resources or `remote` for URLs. Remote resources require Biber. The protocols HTTP and FTP are supported. The remote URL must be a fully qualified path to a `bib` file or a URL which returns a `bib` file.

`type`=*type* default: file

The type of resource. Currently, the only supported type is `file`.

`datatype`=*datatype* default: bibtex

The data type (format) of the resource. The following formats are currently supported:

<code>bibtex</code>	BibTeX format.	
<code>ris</code>	Research Information Systems (RIS) format. ¹⁸ Note that an ID tag is required in all RIS records. The ID value corresponds to the entry key. Support for this format is experimental.	Biber only
<code>zoterordfxml</code>	Zotero RDF/XML format. Support for this format is experimental. Refer to the Biber manual for details.	Biber only

¹⁸[http://en.wikipedia.org/wiki/RIS_\(file_format\)](http://en.wikipedia.org/wiki/RIS_(file_format))

`endnotexml` EndNote XML format. Support for this format is experimental. Refer to the Biber manual for details. Biber only

`\addglobalbib[⟨options⟩]{⟨resource⟩}`

This command differs from `\addbibresource` in that the `⟨resource⟩` is added to the global resource list. The difference between default resources and global resources is only relevant if there are reference sections in the document and the optional argument of `refsection` (§ 3.6.5) is used to specify alternative resources which replace the default resource list. Any global resources are added to all reference sections.

`\addsectionbib[⟨options⟩]{⟨resource⟩}`

This command differs from `\addbibresource` in that the resource `⟨options⟩` are registered but the `⟨resource⟩` not added to any resource list. This is only required for resources which 1) are given exclusively in the optional argument of `refsection` (§ 3.6.5) and 2) require options different from the default settings. In this case, `\addsectionbib` is employed to qualify the `⟨resource⟩` prior to using it by setting the appropriate `⟨options⟩` in the preamble. The `label` option may be useful to assign a short name to the resource.

`\bibliography{⟨bibfile, ...⟩}`

Deprecated

The legacy command for adding bibliographic resources, supported for backwards compatibility. Like `\addbibresource`, this command is only available in the preamble and adds resources to the default resource list. Its argument is a comma-separated list of bib files. The `.bib` extension may be omitted from the filename. Invoking this command multiple times to add more files is permissible. This command is deprecated. Please consider using `\addbibresource` instead.

3.6.2 参考文献表 The Bibliography

`\printbibliography[⟨key=value, ...⟩]`

This command prints the bibliography. It takes one optional argument, which is a list of options given in `⟨key⟩=⟨value⟩` notation. The following options are available:

`env=⟨name⟩` default: bibliography/shorthands

The ‘high-level’ layout of the bibliography and the list of shorthands is controlled by environments defined with `\defbibenvironment`. This option selects an environment. The `⟨name⟩` corresponds to the identifier used when defining the environment with `\defbibenvironment`. By default, the `\printbibliography` command uses the identifier `bibliography`; `\printbiblist` uses `shorthands`. See also §§ 3.6.4 and 3.6.8.

`heading=⟨name⟩` default: bibliography/shorthands

The bibliography and the list of shorthands typically have a chapter or section heading. This option selects the heading `⟨name⟩`, as defined with `\defbibheading`. By default, the `\printbibliography` command uses the heading `bibliography`; `\printbiblist` uses `shorthands`. See also §§ 3.6.4 and 3.6.8.

`title=⟨text⟩`

This option overrides the default title provided by the heading selected with the `heading` option, if supported by the heading definition. See § 3.6.8 for details.

`prenote=⟨name⟩`

The prenote is an arbitrary piece of text to be printed after the heading but before the list of references. This option selects the prenote `⟨name⟩`, as defined with `\defbibnote`. By default, no prenote is printed. The note is printed in the standard text font. It is not affected by `\bibsetup` and `\bibfont` but it may contain its own font declarations. See § 3.6.9 for details.

`postnote=⟨name⟩`

The postnote is an arbitrary piece of text to be printed after the list of references. This option selects the postnote `⟨name⟩`, as defined with `\defbibnote`. By default, no postnote is printed. The note is printed in the standard text font. It is not affected by `\bibsetup` and `\bibfont` but it may contain its own font declarations. See § 3.6.9 for details.

`section=⟨integer⟩` default: current section

Print only entries cited in reference section `⟨integer⟩`. The reference sections are numbered starting at 1. All citations given outside a `refsection` environment are assigned to section 0. See § 3.6.5 for details and § 3.11.3 for usage examples.

`segment=⟨integer⟩` default: 0

Print only entries cited in reference segment `⟨integer⟩`. The reference segments are numbered starting at 1. All citations given outside a `refsegment` environment are assigned to segment 0. See § 3.6.6 for details and § 3.11.3 for usage examples. Remember that segments within a section are numbered local to the section so the segment you request will be the *n*th segment in the requested (or currently active enclosing) section.

`type=⟨entrytype⟩`

Print only entries whose entry type is `⟨entrytype⟩`.

`nottype=⟨entrytype⟩`

Print only entries whose entry type is not `⟨entrytype⟩`. This option may be used multiple times.

`subtype=<subtype>`

Print only entries whose `entrysubtype` is defined and `<subtype>`.

`notsubtype=<subtype>`

Print only entries whose `entrysubtype` is undefined or not `<subtype>`. This option may be used multiple times.

`keyword=<keyword>`

Print only entries whose `keywords` field includes `<keyword>`. This option may be used multiple times.

`notkeyword=<keyword>`

Print only entries whose `keywords` field does not include `<keyword>`. This option may be used multiple times.

`category=<category>`

Print only entries assigned to category `<category>`. This option may be used multiple times.

`notcategory=<category>`

Print only entries not assigned to category `<category>`. This option may be used multiple times.

`filter=<name>`

Filter the entries with filter `<name>`, as defined with `\defbibfilter`. See § 3.6.10 for details.

`check=<name>`

Filter the entries with check `<name>`, as defined with `\defbibcheck`. See § 3.6.10 for details.

`prefixnumbers=<string>`

This option applies to numerical citation/bibliography styles only and requires that the `defernumbers` option from § 3.1.2.1 be enabled globally. Setting this option will implicitly enable `resetnumbers` for the current bibliography. The option assigns the `<string>` as a prefix to all entries in the respective bibliography. For example, if the `<string>` is `A`, the numerical labels printed will be `[A1]`, `[A2]`, `[A3]`, etc. This is useful for subdivided numerical bibliographies where each subbibliography uses a different prefix. The `<string>` is available to styles in the `prefixnumber` field of all affected entries. See § 4.2.4.2 for details.

`resetnumbers=<true,false,number>`

This option applies to numerical citation/bibliography styles only and requires that the `defernumbers` option from § 3.1.2.1 be enabled globally. If enabled, it will reset the numerical labels assigned to the entries in the respective bibliography, i. e., the numbering will restart at 1. You can also pass a number to this option, for example: `resetnumbers=10` to reset numbering to the specified number to aid numbering continuity across documents. Use this option with care as Biblatex can not guarantee unique labels globally if they are reset manually.

`omitnumbers=true, false`

This option applies to numerical citation/bibliography styles only and requires that the `defernumbers` option from § 3.1.2.1 be enabled globally. If enabled, Biblatex will not assign a numerical label to the entries in the respective bibliography. This is useful when mixing a numerical subbibliography with one or more subbibliographies using a different scheme (e. g., author-title or author-year).

`\bibbysection[<key=value, ...>]`

This command automatically loops over all reference sections. This is equivalent to giving one `\printbibliography` command for every section but has the additional benefit of automatically skipping sections without references. Note that `\bibbysection` starts looking for references in section 1. It will ignore references given outside of `refsection` environments since they are assigned to section 0. See § 3.11.3 for usage examples. The options are a subset of those supported by `\printbibliography`. Valid options are `env`, `heading`, `prenote`, `postnote`. The current bibliography context sorting scheme is used for all sections (see § 3.6.11).

`\bibbysegment[<key=value, ...>]`

This command automatically loops over all reference segments. This is equivalent to giving one `\printbibliography` command for every segment in the current `refsection` but has the additional benefit of automatically skipping segments without references. Note that `\bibbysegment` starts looking for references in segment 1. It will ignore references given outside of `refsegment` environments since they are assigned to segment 0. See § 3.11.3 for usage examples. The options are a subset of those supported by `\printbibliography`. Valid options are `env`, `heading`, `prenote`, `postnote`. The current bibliography context sorting scheme is used for all segments (see § 3.6.11).

`\bibbycategory[<key=value, ...>]`

This command loops over all bibliography categories. This is equivalent to giving one `\printbibliography` command for every category but has the additional benefit of automatically skipping empty categories. The categories are processed in the order in which they were declared. See § 3.11.3 for usage examples. The options are a subset

of those supported by `\printbibliography`. Valid options are `env`, `prenote`, `postnote`, `section`. Note that `heading` is not available with this command. The name of the current category is automatically used as the heading name. This is equivalent to passing `heading=\category` to `\printbibliography` and implies that there must be a matching heading definition for every category. The current bibliography context sorting scheme is used for all categories (see § 3.6.11).

`\printbibheading` [*key=value, ...*]

This command prints a bibliography heading defined with `\defbibheading`. It takes one optional argument, which is a list of options given in *key=value* notation. The options are a small subset of those supported by `\printbibliography`. Valid options are `heading` and `title`. By default, this command uses the heading `bibliography`. See § 3.6.8 for details. Also see §§ 3.11.3 and 3.11.4 for usage examples.

To print a bibliography with a different sorting scheme than the global sorting scheme, use the bibliography context switching commands from § 3.6.11.

3.6.3 缩略表 The List of Shorthands

This section applies only to BibTeX. When using Biber, the list of shorthands is just a special case of a bibliography list. See § 3.6.4.¹⁹ BibTeX only

If any entry includes a `shorthand` field, `biblatex` automatically builds a list of shorthands which may be printed in addition to the regular bibliography. The following command prints the list of shorthands.

`\printshorthands` [*key=value, ...*]

This command prints the list of shorthands. It takes one optional argument, which is a list of options given in *key=value* notation. Valid options are all options supported by `\printbibliography` (§ 3.6.2) except `prefixnumbers`, `resetnumbers`, and `omitnumbers`. If there are any `refsection` environments in the document, the list of shorthands will be local to these environments; see § 3.6.5 for details. By default, this command uses the heading `shorthands`. See § 3.6.8 for details.

The `sorting` option differs from `\printbibliography` in that if omitted, the default is to sort by shorthand.

3.6.4 各类参考文献表 Bibliography Lists

This section applies only to Biber. It is a generalisation of the shorthands facility available in earlier versions and with BibTeX. When using BibTeX as the backend, please refer to section § 3.6.3. Biber only

¹⁹本节需要重点关注一下，之前没有搞清楚

Biblatex can, in addition to printing normal bibliographies, also print arbitrary lists of information derived from the bibliography data such as a list of shorthand abbreviations for particular entries or a list of abbreviations of journal titles.

A bibliography list differs from a normal bibliography in that the same bibliography driver is used to print all entries rather than a specific driver being used for each entry depending on the entry type.

```
\printbiblist[⟨key=value, ...⟩]{⟨biblistname⟩}
```

This command prints a bibliography list. It takes an optional argument, which is a list of options given in $\langle key \rangle = \langle value \rangle$ notation. Valid options are all options supported by `\printbibliography` (§ 3.6.2) except `prefixnumbers`, `resetnumbers`, and `omitnumbers`. If there are any `refsection` environments in the document, the bibliography list will be local to these environments; see § 3.6.5 for details. By default, this command uses the heading `biblist`. See § 3.6.8 for details.

The $\langle biblistname \rangle$ is a mandatory argument which names the bibliography list. This name is used to identify:

- The default bibliography driver used to print the list entries
- A default filter declared with `\DeclareBiblistFilter` (see § 4.5.6) used to filter the entries returned from Biber
- A default check declared with `\defbibcheck` (see § 3.6.10) used to post-process the list entries
- The default bib environment to use
- The default sorting scheme name to use

In terms of sorting the list, the default is to sort use the sorting scheme named after the bibliography list (if it exists) and only then to fall back to the current context sorting scheme if this is not defined (see § 3.6.11).

The most common bibliography list is a list of shorthand abbreviations for certain entries and so this has a convenience alias `\printshorthands[...]` for backwards compatibility which is defined as:

```
\printbiblist[...]{shorthand}
```

Biblatex provides automatic support for data source fields in the default data model marked as ‘Label fields’ (See § 2.2.2). Such fields automatically have defined for them:

- A default bib environment (See § 3.6.8)
- A bibliography list filter (See § 4.5.6)
- Some supporting formats and lengths (See § 4.10.5 and § 4.10.4)

Therefore only a minimal setup is required to print bibliography lists with such fields. For example, to print a list of journal title abbreviations, you can minimally put this in your preamble:

```
\DeclareBibliographyDriver{shortjournal}{%
  \printfield{journaltitle}}
```

Then you can put this in your document where you want to print the list:

```
\printbiblist[title={Journal Shorthands}]{shortjournal}
```

Since `shortjournal` is defined in the default data model as a ‘Label field’, this example:

- Uses the automatically created ‘shortjournal’ bib environment
- Uses the automatically created ‘shortjournal’ bibliography list filter to return only entries with a `shortjournal` field in the `.bbl`
- Uses the defined ‘shortjournal’ bibliography driver to print the entries
- Uses the default ‘biblist’ heading but overrides the title with ‘Journal Shorthands’
- Uses the current bibliography context sorting scheme no scheme exists with the name `shortjournal`

Often, you will want to sort on the label field of the list and since a sorting scheme is automatically picked up if it is named after the list, in this case you could simply do:

```
\DeclareSortingScheme{shortjournal}{
  \sort{
    \field{shortjournal}
  }
}
```

Naturally all defaults can be overridden by options to `\printbiblist` and definitions of the environments, filters etc. and in this way arbitrary types of bibliography lists can be printed containing a variety of information from the bibliography data.

Bibliography lists are often used to print lists of various kinds of shorthands and this can result in duplicate entries if more than one bibliography entry has the same shorthand. For example, several journal articles in the same journal would result in duplicate entries in a list of journal shorthands. You can use the fact that such lists automatically pick up a `\bibcheck` with the same name as the list to define a check to remove duplicates. If you are defining a list to print all of the journal shorthands using the `shortjournal` field, you could define a `\bibcheck` like this:

```
\defbibcheck{shortjournal}{%
  \iffieldundef{shortjournal}{\skipentry}{%
    \iffieldundef{journal}{\skipentry}{%
      \ifcsdef{\strfield{shortjournal}=\strfield{journal}}
        {\skipentry}
      {\savefieldcs{journal}{\strfield{shortjournal}=\strfield{journal}}}}}}
```

3.6.5 参考文献分节 Bibliography Sections

The `refsection` environment is used in the document body to mark a reference section. This environment is useful if you want separate, independent bibliographies and bibliography lists in each chapter, section, or any other part of a document. Within a reference section, all cited works are assigned labels which are local to the environment. Technically, reference sections are completely independent from document divisions such as `\chapter` and `\section` even though they will most likely be used per chapter or section. See the `refsection` package option in § 3.1.2.1 for a way to automate this. Also see § 3.11.3 for usage examples.

```
\begin{refsection}[\langle resource, ... \rangle]
\end{refsection}
```

The optional argument is a comma-separated list of resources specific to the reference section. If the argument is omitted, the reference section will use the default resource list, as specified with `\addbibresource` in the preamble. If the argument is provided, it replaces the default resource list. Global resources specified with `\addglobalbib` are always considered. `refsection` environments may not be nested, but you may use `refsegment` environments within a `refsection` to subdivide it into segments. Use the `section` option of `\printbibliography` to select a section when printing the bibliography, and the corresponding option of `\printbiblist` when printing bibliography lists. Bibliography sections are numbered starting at 1. The number of the current section is also written to the transcript file. All citations given outside a `refsection` environment are assigned to section 0. If `\printbibliography` is used within a `refsection`, it will automatically select the current section. The `section` option is not required in this case. This also applies to `\printbiblist`.

```
\newrefsection[\langle resource, ... \rangle]
```

This command is similar to the `refsection` environment except that it is a stand-alone command rather than an environment. It automatically ends the previous reference section (if any) and immediately starts a new one. Note that the reference section started by the last `\newrefsection` command in the document will extend to the very end of the document. Use `\endrefsection` if you want to terminate it earlier.

3.6.6 参考文献分部 Bibliography Segments

The `refsegment` environment is used in the document body to mark a reference segment. This environment is useful if you want one global bibliography which is subdivided by chapter, section, or any other part of the document. Technically, reference segments are completely independent from document divisions such as `\chapter` and `\section` even though they will typically be used per chapter or section. See the `refsegment` package option in § 3.1.2.1 for a way to automate this. Also see § 3.11.3 for usage examples.

```
\begin{refsegment}

\end{refsegment}
```

The difference between a `refsection` and a `refsegment` environment is that the former creates labels which are local to the environment whereas the latter provides a target for the `segment` filter of `\printbibliography` without affecting the labels. They will be unique across the entire document. `refsegment` environments may not be nested, but you may use them in conjunction with `refsection` to subdivide a reference section into segments. In this case, the segments are local to the enclosing `refsection` environment. Use the `segment` option of `\printbibliography` to select a segment when printing the bibliography. Within a section, the reference segments are numbered starting at 1 and the number of the current segment will be written to the transcript file. All citations given outside a `refsegment` environment are assigned to segment 0. In contrast to the `refsection` environment, the current segment is not selected automatically if `\printbibliography` is used within a `refsegment` environment.

`\newrefsegment` This command is similar to the `refsegment` environment except that it is a stand-alone command rather than an environment. It automatically ends the previous reference segment (if any) and immediately starts a new one. Note that the reference segment started by the last `\newrefsegment` command will extend to the end of the document. Use `\endrefsegment` if you want to terminate it earlier.

3.6.7 参考文献分类 Bibliography Categories

Bibliography categories allow you to split the bibliography into multiple parts dedicated to different topics or different types of references, for example primary and secondary sources. See § 3.11.4 for usage examples.

```
\DeclareBibliographyCategory{<category>}
```

Declares a new `<category>`, to be used in conjunction with `\addtocategory` and the `category` and `notcategory` filters of `\printbibliography`. This command is used in the document preamble.

```
\addtocategory{<category>}{<key>}
```

Assigns a `<key>` to a `<category>`, to be used in conjunction with the `category` and `notcategory` filters of `\printbibliography`. This command may be used in the preamble and in the document body. The `<key>` may be a single entry key or a comma-separated list of keys. The assignment is global.

3.6.8 参考文献标题和环境 Bibliography Headings and Environments

```
\defbibenvironment{<name>}{<begin code>}{<end code>}{<item code>}
```

This command defines bibliography environments. The `<name>` is an identifier passed to the `env` option of `\printbibliography` and `\printbiblist` when selecting

the environment. The *begin code* is LaTeX code to be executed at the beginning of the environment; the *end code* is executed at the end of the environment; the *item code* is code to be executed at the beginning of each entry in the bibliography or a bibliography list. Here is an example of a definition based on the standard LaTeX `list` environment:

```
\defbibenvironment{bibliography}
  {\list{}
    {\setlength{\leftmargin}{\bibhang}%
     \setlength{\itemindent}{-\leftmargin}%
     \setlength{\itemsep}{\bibitemsep}%
     \setlength{\parsep}{\bibparsep}}}
  {\endlist}
  {\item}
```

As seen in the above example, usage of `\defbibenvironment` is roughly similar to `\newenvironment` except that there is an additional mandatory argument for the *item code*.

`\defbibheading{<name>}[<title>]{<code>}`

This command defines bibliography headings. The *name* is an identifier to be passed to the heading option of `\printbibliography` or `\printbibheading` and `\printbiblist` when selecting the heading. The *code* should be LaTeX code generating a fully-fledged heading, including page headers and an entry in the table of contents, if desired. If `\printbibliography` or `\printbiblist` are invoked with a *title* option, the title will be passed to the heading definition as #1. If not, the default title specified by the optional *title* argument is passed as #1 instead. The *title* argument will typically be `\bibname`, `\refname`, or `\biblistname` (see § 4.9.2.1). This command is often needed after changes to document headers in the preamble. Here is an example of a simple heading definition:

```
\defbibheading{bibliography}[\bibname]{%
  \chapter*{#1}%
  \markboth{#1}{#1}}
```

The following headings, which are intended for use with `\printbibliography` and `\printbibheading`, are predefined:

`bibliography`

This is the default heading used by `\printbibliography` if the heading option is not given. Its default definition depends on the document class. If the class provides a `\chapter` command, the heading is similar to the bibliography heading of the

standard LaTeX book class, i.e., it uses `\chapter*` to create an unnumbered chapter heading which is not included in the table of contents. If there is no `\chapter` command, it is similar to the bibliography heading of the standard LaTeX article class, i.e., it uses `\section*` to create an unnumbered section heading which is not included in the table of contents. The string used in the heading also depends on the document class. With book-like classes the localization string `bibliography` is used, with other classes it is `references` (see § 4.9.2). See also §§ 3.12.1 and 3.12.2 for class-specific hints.

`subbibliography`

Similar to `bibliography` but one sectioning level lower. This heading definition uses `\section*` instead of `\chapter*` with a book-like class and `\subsection*` instead of `\section*` otherwise.

`bibintoc`

Similar to `bibliography` above but adds an entry to the table of contents.

`subbibintoc`

Similar to `subbibliography` above but adds an entry to the table of contents.

`bibnumbered`

Similar to `bibliography` above but uses `\chapter` or `\section` to create a numbered heading which is also added to the table of contents.

`subbibnumbered`

Similar to `subbibliography` above but uses `\section` or `\subsection` to create a numbered heading which is also added to the table of contents.

`none`

A blank heading definition. Use this to suppress the heading.

The following headings intended for use with `\printbiblist` are predefined:

`biblist`

This is the default heading used by `\printbiblist` if the `heading` option is not given. It is similar to `bibliography` above except that it uses the localization string `shorthands` instead of `bibliography` or `references` (see § 4.9.2). See also §§ 3.12.1 and 3.12.2 for class-specific hints.

`biblistintoc`

Similar to `biblist` above but adds an entry to the table of contents.

`biblistnumbered`

Similar to `biblist` above but uses `\chapter` or `\section` to create a numbered heading which is also added to the table of contents.

3.6.9 参考文献注释 Bibliography Notes

`\defbibnote{⟨name⟩}{⟨text⟩}`

Defines the bibliography note `⟨name⟩`, to be used via the `prenote` and `postnote` options of `\printbibliography` and `\printbiblist`. The `⟨text⟩` may be any arbitrary piece of text, possibly spanning several paragraphs and containing font declarations. Also see § 3.12.6.

3.6.10 参考文献过滤和检查 Bibliography Filters and Checks

`\defbibfilter{⟨name⟩}{⟨expression⟩}`

Defines the custom bibliography filter `⟨name⟩`, to be used via the `filter` option of `\printbibliography`. The `⟨expression⟩` is a complex test based on the logical operators `and`, `or`, `not`, the group separator `(. . .)`, and the following atomic tests:

`segment=⟨integer⟩`

Matches all entries cited in reference segment `⟨integer⟩`.

`type=⟨entrytype⟩`

Matches all entries whose entry type is `⟨entrytype⟩`.

`subtype=⟨subtype⟩`

Matches all entries whose `entrysubtype` is `⟨subtype⟩`.

`keyword=⟨keyword⟩`

Matches all entries whose `keywords` field includes `⟨keyword⟩`. If the `⟨keyword⟩` contains spaces, it needs to be wrapped in braces.

`category=⟨category⟩`

Matches all entries assigned to `⟨category⟩` with `\addtocategory`.

Here is an example of a filter expression:

```
\defbibfilter{example}{%
  ( type=book or type=inbook )
  and keyword=abc
  and not keyword={x y z}
}
```


This filter will match all entries whose entry type is either @book or @inbook and whose keywords field includes the keyword ‘abc’ but not ‘x y z’. As seen in the above example, all elements are separated by whitespace (spaces, tabs, or line endings). There is no spacing around the equal sign. The logical operators are evaluated with the `\ifboolexpr` command from the `etoolbox` package. See the `etoolbox` manual for details about the syntax. The syntax of the `\ifthenelse` command from the `ifthen` package, which has been employed in older versions of Bibl_{at}ex, is still supported. This is the same test using `ifthen`-like syntax:

```
\defbibfilter{example}{%
  \(\type{book} \or \type{inbook} \)
  \and \keyword{abc}
  \and \not \keyword{x y z}
}
```

Note that custom filters are local to the reference section in which they are used. Use the section filter of `\printbibliography` to select a different section. This is not possible from within a custom filter.

```
\defbibcheck{<name>}{<code>}
```

Defines the custom bibliography filter *<name>*, to be used via the `check` option of `\printbibliography`. `\defbibcheck` is similar in concept to `\defbibfilter` but much more low-level. Rather than a high-level expression, the *<code>* is LaTeX code, much like the code used in driver definitions, which may perform arbitrary tests to decide whether or not a given entry is to be printed. The bibliographic data of the respective entry is available when the *<code>* is executed. Issuing the command `\skipentry` in the *<code>* will cause the current entry to be skipped. For example, the following filter will only output entries with an `abstract` field:

```
\defbibcheck{abstract}{%
  \iffieldundef{abstract}{\skipentry}{}
  ...
\printbibliography[check=abstract]
```

The following check will exclude all entries published before the year 2000:

```
\defbibcheck{recent}{%
  \iffieldint{year}
    {\ifnumless{\thefield{year}}{2000}
      {\skipentry}
      {}}
  {\skipentry}}
```

See the author guide, in particular §§ 4.6.2 and 4.6.3, for further details.

3.6.11 参考文献表内容 Bibliography Contexts

References in a bibliography are cited and printed in a ‘context’. The context determines the data which is actually used for an entry. A context consists currently of the following information (the ‘context’ concept is designed for future extensibility):

- A specific sorting scheme

The data for an entry can vary depending on the context. This is most obvious in the case of the `extra*` fields like `extrayear` which are generated by the backend according to the order of entries *after* sorting so that they come out in the expected ‘a, b, c’ order. This clearly shows that the *data* in an entry can be different between sorting schemes. If a document contains more than one bibliography list with different sorting schemes, it can happen then that the `.bbl` contains sorting lists with the same entry but containing different data (a different value for `extrayear`, for example). The purpose of bibliography contexts is to encapsulate things inside a context so that Bibl_{at}ex can use the correct entry data. An example is printing a bibliography list with a different sorting order to the global sorting order:

```
\usepackage[sorting=nyt]{biblatex}
\cite{one}
\cite{two}
\printbibliography
\newrefcontext[sorting=yndt]
\cite{one}
\cite{two}
\printbibliography
```

Here, the second use of the citations, along with the `\printbibliography` command will use data from the context of the ‘yndt’ sorting scheme which may well be different from the data associated with the default ‘nyt’ scheme.

```
\begin{refcontext}[<key=value, ...>]
\end{refcontext}
```

Wraps a bibliography context environment. The options define the context attributes. All context attributes are optional and default to the global settings if absent. The current options are:

`sorting=<name>`

Specify a sorting scheme defined previously with `\DeclareSortingScheme`. This scheme is used to determine which data to retrieve and/or print for an entry in the commands inside the context.

The `refcontext` environment cannot be nested and Bibl_{at}ex will generate an error if you try to do so.

`\newrefcontext[⟨key=value, ...⟩]`

This command is similar to the `refcontext` environment except that it is a stand-alone command rather than an environment. It automatically ends the previous context section (if any) and immediately starts a new one. Note that the context section started by the last `\newrefcontext` command in the document will extend to the very end of the document. Use `\endrefcontext` if you want to terminate it earlier.

At the beginning of the document, there is always a global context containing global settings for each of the context attributes.

3.6.12 动态条目集 *Dynamic Entry Sets*

In addition to the `@set` entry type, Biblatex also supports dynamic entry sets defined on a per-document/per-refsection basis. The following command, which may be used in the document preamble or the document body, defines the set `⟨key⟩`:

`\defbibentryset{⟨key⟩}{⟨key1,key2,key3, ...⟩}`

Biber only

The `⟨key⟩` is the entry key of the set, which is used like any other entry key when referring to the set. The `⟨key⟩` must be unique and it must not conflict with any other entry key. The second argument is a comma-separated list of the entry keys which make up the set. `\defbibentryset` implies the equivalent of a `\nocite` command, i. e., all sets which are declared are also added to the bibliography. When declaring the same set more than once, only the first invocation of `\defbibentryset` will define the set. Subsequent definitions of the same `⟨key⟩` are ignored and work like `\nocite⟨key⟩`. Dynamic entry sets defined in the document body are local to the enclosing `refsection` environment, if any. Otherwise, they are assigned to reference section 0. Those defined in the preamble are assigned to reference section 0. Note that dynamic entry sets require Biber. They will not work with any other backend. See § 3.11.5 for further details.

3.7 引用命令 *Citation Commands*

All citation commands generally take one mandatory and two optional arguments. The `⟨prenote⟩` is text to be printed at the beginning of the citation. This is usually a notice such as ‘see’ or ‘compare’. The `⟨postnote⟩` is text to be printed at the very end of the citation. This is usually a page number. If only one of these arguments is given, it is taken as a postnote. If you want to specify a prenote but no postnote, you need to leave the second optional argument empty, as in `\cite[see][]{key}`. The `⟨key⟩` argument to all citation commands is mandatory. This is the entry key or a comma-separated list of keys corresponding to the entry keys in the `bib` file. In sum, all basic citations commands listed further down have the following syntax:

`\command[⟨prenote⟩][⟨postnote⟩]{⟨keys⟩}⟨punctuation⟩`

If the `autopunct` package option from § 3.1.2.1 is enabled, they will scan ahead for any `⟨punctuation⟩` immediately following their last argument. This is useful to avoid spurious punctuation marks after citations. This feature is configured with `\DeclareAutoPunctuation`, see § 4.7.5 for details.

3.7.1 标准命令 Standard Commands

The following commands are defined by the citation style. Citation styles may provide any arbitrary number of specialized commands, but these are the standard commands typically provided by general-purpose styles.

`\cite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}`

`\Cite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}`

These are the bare citation commands. They print the citation without any additions such as parentheses. The numeric and alphabetic styles still wrap the label in square brackets since the reference may be ambiguous otherwise. `\Cite` is similar to `\cite` but capitalizes the name prefix of the first name in the citation if the `useprefix` option is enabled, provided that there is a name prefix and the citation style prints any name at all.

`\parencite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}`

`\Parencite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}`

These commands use a format similar to `\cite` but enclose the entire citation in parentheses. The numeric and alphabetic styles use square brackets instead. `\Parencite` is similar to `\parencite` but capitalizes the name prefix of the first name in the citation if the `useprefix` option is enabled, provided that there is a name prefix and the citation style prints any name at all.

`\footcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}`

`\footcitetext[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}`

These command use a format similar to `\cite` but put the entire citation in a footnote and add a period at the end. In the footnote, they automatically capitalize the name prefix of the first name if the `useprefix` option is enabled, provided that there is a name prefix and the citation style prints any name at all. `\footcitetext` differs from `\footcite` in that it uses `\footnotetext` instead of `\footnote`.

3.7.2 样式相关命令 Style-specific Commands

The following additional citation commands are only provided by some of the citation styles which ship with this package.

```
\textcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\Textcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

These citation commands are provided by all styles that ship with this package. They are intended for use in the flow of text, replacing the subject of a sentence. They print the authors or editors followed by a citation label which is enclosed in parentheses. Depending on the citation style, the label may be a number, the year of publication, an abridged version of the title, or something else. The numeric and alphabetic styles use square brackets instead of parentheses. In the verbose styles, the label is provided in a footnote. Trailing punctuation is moved between the author or editor names and the footnote mark. `\Textcite` is similar to `\textcite` but capitalizes the name prefix of the first name in the citation if the `useprefix` option is enabled, provided that there is a name prefix.

```
\smartcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\Smartcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

Like `\parencite` in a footnote and like `\footcite` in the body.

```
\cite*[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

This command is provided by all author-year and author-title styles. It is similar to the regular `\cite` command but merely prints the year or the title, respectively.

```
\parencite*[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

This command is provided by all author-year and author-title styles. It is similar to the regular `\parencite` command but merely prints the year or the title, respectively.

```
\supercite{⟨key⟩}
```

This command, which is only provided by the numeric styles, prints numeric citations as superscripts without brackets. It uses `\supercitedelim` instead of `\multicitedelim` as citation delimiter. Note that any `⟨prenote⟩` and `⟨postnote⟩` arguments are ignored. If they are given, `\supercite` will discard them and issue a warning message.

3.7.3 有限标注表 Qualified Citation Lists

This package supports a class of special citation commands called ‘multicite’ commands. The point of these commands is that their argument is a list of citations where each item forms a fully qualified citation with a pre- and/or postnote. This is particularly useful with parenthetical citations and citations given in footnotes. It is also possible to assign a pre- and/or postnote to the entire list. The multicite commands are built on top of backend commands like `\parencite` and `\footcite`. The citation style provides a multicite definition with `\DeclareMultiCiteCommand` (see § 4.3.1). The following example illustrates the syntax of multicite commands:

```
\parencites[35]{key1}[88--120]{key2}[23]{key3}
```

The format of the arguments is similar to that of the regular citation commands, except that only one citation command is given. If only one optional argument is given for an item in the list, it is taken as a postnote. If you want to specify a prenote but no postnote, you need to leave the second optional argument of the respective item empty:

```
\parencites[35]{key1}[chapter 2 in][]{key2}[23]{key3}
```

In addition to that, the entire citation list may also have a pre- and/or postnote. The syntax of these global notes differs from other optional arguments in that they are given in parentheses rather than the usual brackets:

```
\parencites(and chapter 3)[35]{key1}[78]{key2}[23]{key3}
\parencites(Compare)()[35]{key1}[78]{key2}[23]{key3}
\parencites(See)(and the introduction)[35]{key1}[78]{key2}[23]{key3}
```

Note that the multicite commands keep on scanning for arguments until they encounter a token that is not the start of an optional or mandatory argument. If a left brace or bracket follows a multicite command, you need to mask it by adding `\relax` or a control space (a backslash followed by a space) after the last valid argument. This will cause the scanner to stop.

```
\parencites[35]{key1}[78]{key2}\relax[...]
\parencites[35]{key1}[78]{key2}\_{}{...}
```

By default, this package provides the following multicite commands which correspond to regular commands from §§ 3.7.1 and 3.7.2:

```
\cites(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
\Cites(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
```

The multicite version of `\cite` and `\Cite`, respectively.

```
\parencites(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
\Parencites(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
```

The multicite version of `\parencite` and `\Parencite`, respectively.

```
\footcites(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
\footcitetexts(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
```

The multicite version of `\footcite` and `\footcitetext`, respectively.

```
\smartcites(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
\Smartcites(<multiprenote>)(<multi postnote>)[<prenote>][<postnote>]{<key>}...[<prenote>][<postnote>]{<key>}
```

The multicite version of `\smartcite` and `\Smartcite`, respectively.

```
\textcites (<multiprenote>) (<multi postnote>) [<prenote>] [<postnote>] {<key>} . . . [<prenote>] [<postnote>] {<key>}
\Textcites (<multiprenote>) (<multi postnote>) [<prenote>] [<postnote>] {<key>} . . . [<prenote>] [<postnote>] {<key>}
```

The multicite version of `\textcite` and `\Textcite`, respectively.

```
\supercites (<multiprenote>) (<multi postnote>) [<prenote>] [<postnote>] {<key>} . . . [<prenote>] [<postnote>] {<key>}
```

The multicite version of `\supercite`. This command is only provided by the numeric styles.

3.7.4 与样式无关的命令 **Style-independent Commands**

Sometimes it is desirable to give the citations in the source file in a format that is not tied to a specific citation style and can be modified globally in the preamble. The format of the citations is easily changed by loading a different citation style. However, when using commands such as `\parencite` or `\footcite`, the way the citations are integrated with the text is still effectively hard-coded. The idea behind the `\autocite` command is to provide higher-level citation markup which makes global switching from inline citations to citations given in footnotes (or as superscripts) possible. The `\autocite` command is built on top of backend commands like `\parencite` and `\footcite`. The citation style provides an `\autocite` definition with `\DeclareAutoCiteCommand` (see § 4.3.1). This definition may be activated with the `autocite` package option from § 3.1.2.1. The citation style will usually initialize this package option to a value which is suitable for the style, see § 3.3.1 for details. Note that there are certain limits to high-level citation markup. For example, inline author-year citation schemes often integrate citations so tightly with the text that it is virtually impossible to automatically convert them to footnotes. The `\autocite` command is only applicable in cases in which you would normally use `\parencite` or `\footcite` (or `\supercite`, with a numeric style). The citations should be given at the end of a sentence or a partial sentence, immediately preceding the terminal punctuation mark, and they should not be a part of the sentence in a grammatical sense (like `\textcite`, for example).

```
\autocite [<prenote>] [<postnote>] {<key>}
\Autocite [<prenote>] [<postnote>] {<key>}
```

In contrast to other citation commands, the `\autocite` command does not only scan ahead for punctuation marks following its last argument to avoid double punctuation marks, it actually moves them around if required. For example, with `autocite=footnote`, a trailing punctuation mark will be moved such that the footnote mark is printed after the punctuation. `\Autocite` is similar to `\autocite` but capitalizes the name prefix of the first name in the citation if the `useprefix` option is enabled, provided that there is a name prefix and the citation style prints any name at all.

```
\autocite* [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
\Autocite* [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
```

The starred variants of `\autocite` do not behave differently from the regular ones. The asterisk is simply passed on to the backend command. For example, if `\autocite` is configured to use `\parencite`, then `\autocite*` will execute `\parencite*`.

```
\autocites (⟨multiprenote⟩) (⟨multipostnote⟩) [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩} . . . [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
\Autocites (⟨multiprenote⟩) (⟨multipostnote⟩) [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩} . . . [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
```

This is the multicite version of `\autocite`. It also detects and moves punctuation if required. Note that there is no starred variant. `\Autocites` is similar to `\autocites` but capitalizes the name prefix of the first name in the citation if the `useprefix` option is enabled, provided that there is a name prefix and the citation style prints any name at all.

3.7.5 文本命令 Text Commands

The following commands are provided by the core of Bibl_{at}ex. They are intended for use in the flow of text. Note that all text commands are excluded from citation tracking.

```
\citeauthor [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
\citeauthor* [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
\Citeauthor [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
\Citeauthor* [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
```

These commands print the authors. Strictly speaking, it prints the `labelname` list, which may be the author, the editor, or the translator. `\Citeauthor` is similar to `\citeauthor` but capitalizes the name prefix of the first name in the citation if the `useprefix` option is enabled, provided that there is a name prefix. The starred variants effectively force `maxcitenames` to 1 for just this command on so only print the first name in the `labelname` list (potentially followed by the “et al” string if there are more names). This allows more natural textual flow when referring to a paper in the singular when otherwise `\citeauthor` would generate a (naturally plural) list of names.

```
\citetitle [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
\citetitle* [⟨prenote⟩] [⟨postnote⟩] {⟨key⟩}
```

This command prints the title. It will use the abridged title in the `shorttitle` field, if available. Otherwise it falls back to the full title found in the `title` field. The starred variant always prints the full title.


```
\citeyear[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}  
\citeyear*[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

This command prints the year (year field or year component of date). The starred variant includes the extrayear information, if any.

```
\citedate[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}  
\citedate*[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

This command prints the full date (date or year). The starred variant includes the extrayear information, if any.

```
\citeurl[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

This command prints the url field.

```
\parentext{⟨text⟩}
```

This command wraps the *⟨text⟩* in context sensitive parentheses.

```
\brackettext{⟨text⟩}
```

This command wraps the *⟨text⟩* in context sensitive brackets.

3.7.6 特殊命令 Special Commands

The following special commands are also provided by the core of Biblatex.

```
\nocite{⟨key⟩}  
\nocite{*}
```

This command is similar to the standard LaTeX `\nocite` command. It adds the *⟨key⟩* to the bibliography without printing a citation. If the *⟨key⟩* is an asterisk, all entries available in the bib file are added to the bibliography. Like all other citation commands, `\nocite` commands in the document body are local to the enclosing `refsection` environment, if any. In contrast to standard LaTeX, `\nocite` may also be used in the document preamble. In this case, the references are assigned to reference section 0.

```
\fullcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

This command uses the bibliography driver for the respective entry type to create a full citation similar to the bibliography entry. It is thus related to the bibliography style rather than the citation style.

```
\footfullcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
```

Similar to `\fullcite` but puts the entire citation in a footnote and adds a period at the end.

```
\volcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Volcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
```

These commands are similar to `\cite` and `\Cite` but intended for references to multi-volume works which are cited by volume and page number. Instead of the `⟨postnote⟩`, they take a mandatory `⟨volume⟩` and an optional `⟨page⟩` argument. Since they merely compose the postnote and pass it to the `\cite` command provided by the citation style as a `⟨postnote⟩` argument, these commands are style independent. The format of the volume portion is controlled by the field formatting directive `volcitevolume`, the format of the page/text portion is controlled by the field formatting directive `volcitepages` (§ 4.10.4). The delimiter printed between the volume portion and the page/text portion may be modified by redefining the macro `\volcitedelim` (§ 4.10.1).

```
\volcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
... [⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Volcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
... [⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
```

The multicite version of `\volcite` and `\Volcite`, respectively.

```
\pvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Pvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
```

Similar to `\volcite` but based on `\parencite`.

```
\pvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
... [⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Pvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
... [⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
```

The multicite version of `\pvolcite` and `\Pvolcite`, respectively.

```
\fvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\ftvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
```

Similar to `\volcite` but based on `\footcite` and `\footcitetext`, respectively.

```
\fvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
... [⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Fvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
... [⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
```

The multicite version of `\fvolcite` and `\Fvolcite`, respectively.

```
\svolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Svolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
```

Similar to `\volcite` but based on `\smartcite`.

```

\svolcites (<multiprenote>) (<multiupostnote>) [<prenote>] {<volume>} [<page>] {<key>}
... [<prenote>] {<volume>} [<page>] {<key>}
\Svolcites (<multiprenote>) (<multiupostnote>) [<prenote>] {<volume>} [<page>] {<key>}
... [<prenote>] {<volume>} [<page>] {<key>}

```

The multicite version of \svolcite and \Svolcite, respectively.

```

\tvolcite [<prenote>] {<volume>} [<page>] {<key>}
\Tvolcite [<prenote>] {<volume>} [<page>] {<key>}

```

Similar to \volcite but based on \textcite.

```

\tvolcites (<multiprenote>) (<multiupostnote>) [<prenote>] {<volume>} [<page>] {<key>}
... [<prenote>] {<volume>} [<page>] {<key>}
\Tvolcites (<multiprenote>) (<multiupostnote>) [<prenote>] {<volume>} [<page>] {<key>}
... [<prenote>] {<volume>} [<page>] {<key>}

```

The multicite version of \tvollcite and \Tvolcite, respectively.

```

\avolcite [<prenote>] {<volume>} [<page>] {<key>}
\Avolcite [<prenote>] {<volume>} [<page>] {<key>}

```

Similar to \volcite but based on \autocite.

```

\avolcites (<multiprenote>) (<multiupostnote>) [<prenote>] {<volume>} [<page>] {<key>}
... [<prenote>] {<volume>} [<page>] {<key>}
\Avolcites (<multiprenote>) (<multiupostnote>) [<prenote>] {<volume>} [<page>] {<key>}
... [<prenote>] {<volume>} [<page>] {<key>}

```

The multicite version of \avolcite and \Avolcite, respectively.

```

\notecite [<prenote>] [<postnote>] {<key>}
\Notecite [<prenote>] [<postnote>] {<key>}

```

These commands print the <prenote> and <postnote> arguments but no citation. Instead, a \nocite command is issued for every <key>. This may be useful for authors who incorporate implicit citations in their writing, only giving information not mentioned before in the running text, but who still want to take advantage of the automatic <postnote> formatting and the implicit \nocite function. This is a generic, style-independent citation command. Special citation styles may provide smarter facilities for the same purpose. The capitalized version forces capitalization (note that this is only applicable if the note starts with a command which is sensitive to Biblalex's punctuation tracker).

```

\pnotecite [<prenote>] [<postnote>] {<key>}
\Pnotecite [<prenote>] [<postnote>] {<key>}

```

Similar to \notecite but the notes are printed in parentheses.

`\fnotecite`[$\langle prenote \rangle$][$\langle postnote \rangle$]{ $\langle key \rangle$ }

Similar to `\notecite` but the notes are printed in a footnote.

3.7.7 底层命令 Low-level Commands

The following commands are also provided by the core of Biblatex. They grant access to all lists and fields at a lower level.

`\citename`[$\langle prenote \rangle$][$\langle postnote \rangle$]{ $\langle key \rangle$ }[$\langle format \rangle$]{ $\langle name list \rangle$ }

The $\langle format \rangle$ is a formatting directive defined with `\DeclareNameFormat`. Formatting directives are discussed in § 4.4.2. If this optional argument is omitted, this command falls back to the format `citename`. The last argument is the name of a $\langle name list \rangle$, in the sense explained in § 2.2.

`\citelist`[$\langle prenote \rangle$][$\langle postnote \rangle$]{ $\langle key \rangle$ }[$\langle format \rangle$]{ $\langle literal list \rangle$ }

The $\langle format \rangle$ is a formatting directive defined with `\DeclareListFormat`. Formatting directives are discussed in § 4.4.2. If this optional argument is omitted, this command falls back to the format `citelist`. The last argument is the name of a $\langle literal list \rangle$, in the sense explained in § 2.2.

`\citefield`[$\langle prenote \rangle$][$\langle postnote \rangle$]{ $\langle key \rangle$ }[$\langle format \rangle$]{ $\langle field \rangle$ }

The $\langle format \rangle$ is a formatting directive defined with `\DeclareFieldFormat`. Formatting directives are discussed in § 4.4.2. If this optional argument is omitted, this command falls back to the format `citefield`. The last argument is the name of a $\langle field \rangle$, in the sense explained in § 2.2.

3.7.8 其它命令 Miscellaneous Commands

The commands in this section are little helpers related to citations.

`\citereset` This command resets the citation style. This may be useful if the style replaces repeated citations with abbreviations like *ibidem*, *idem*, *op. cit.*, etc. and you want to force a full citation at the beginning of a new chapter, section, or some other location. The command executes a style specific initialization hook defined with the `\InitializeCitationStyle` command from § 4.3.1. It also resets the internal citation trackers of this package. The reset will affect the `\ifciteseen`, `\ifentryseen`, `\ifciteibid`, and `\ifciteidem` tests discussed in § 4.6.2. When used inside a `refsection` environment, the reset of the citation tracker is local to the current `refsection` environment. Also see the `citereset` package option in § 3.1.2.1.

`\citereset*` Similar to `\citereset` but only executes the style's initialization hook, without resetting the internal citation trackers.

`\mancite` Use this command to mark manually inserted citations if you mix automatically generated and manual citations. This is particularly useful if the citation style replaces repeated citations by an abbreviation like *ibidem* which may get ambiguous or misleading otherwise. Always use `\mancite` in the same context as the manual citation, e. g., if the citation is given in a footnote, include `\mancite` in the footnote. The `\mancite` command executes a style specific reset hook defined with the `\OnManualCitation` command from § 4.3.1. It also resets the internal ‘ibidem’ and ‘idem’ trackers of this package. The reset will affect the `\ifciteibid` and `\ifciteidem` tests discussed in § 4.6.2.

`\pno` This command forces a single page prefix in the $\langle postnote \rangle$ argument to a citation command. See § 3.12.3 for further details and usage instructions. Note that this command is only available locally in citations and the bibliography.

`\ppno` Similar to `\pno` but forces a range prefix. See § 3.12.3 for further details and usage instructions. Note that this command is only available locally in citations and the bibliography.

`\nopp` Similar to `\pno` but suppresses all prefixes. See § 3.12.3 for further details and usage instructions. Note that this command is only available locally in citations and the bibliography.

`\psq` In the $\langle postnote \rangle$ argument to a citation command, this command indicates a range of two pages where only the starting page is given. See § 3.12.3 for further details and usage instructions. The suffix printed is the localization string *sequens*, see § 4.9.2. The spacing inserted between the suffix and the page number may be modified by redefining the macro `\sqspace`. The default is an unbreakable interword space. Note that this command is only available locally in citations and the bibliography.

`\psqq` Similar to `\psq` but indicates an open-ended page range. See § 3.12.3 for further details and usage instructions. The suffix printed is the localization string *sequentes*, see § 4.9.2. This command is only available locally in citations and the bibliography.

`\RN{ $\langle integer \rangle$ }`

This command prints an integer as an uppercase Roman numeral. The formatting applied to the numeral may be modified by redefining the macro `\RNfont`.

`\Rn{ $\langle integer \rangle$ }`

Similar to `\RN` but prints a lowercase Roman numeral. The formatting applied to the numeral may be modified by redefining the macro `\Rnfont`.

3.7.9 与natbib兼容的命令 natbib Compatibility Commands

The `natbib` package option loads a `natbib` compatibility module. The module defines aliases for the citation commands provided by the `natbib` package. This includes aliases for the core citation commands `\citet` and `\citep` as well as the

variants `\citealt` and `\citealp`. The starred variants of these commands, which print the full author list, are also supported. The `\cite` command, which is handled in a particular way by `natbib`, is not treated in a special way. The text commands (`\citeauthor`, `\citeyear`, etc.) are also supported, as are all commands which capitalize the name prefix (`\Citet`, `\Citep`, `\Citeauthor`, etc.). Aliasing with `\defcitealias`, `\citetalias`, and `\citepalias` is possible as well. Note that the compatibility commands will not emulate the citation format of the `natbib` package. They merely alias `natbib`'s commands to functionally equivalent facilities of the Biblatex package. The citation format depends on the main citation style. However, the compatibility style will adapt `\nameyear delim` to match the default style of the `natbib` package.

3.7.10 类似`mcite`给的标注命令

Biber only

The `mcite` package option loads a special citation module which provides `mcite`/`mciteplus`-like citation commands. Strictly speaking, what the module provides are wrappers for the commands of the main citation style. For example, the following command:

```
\mcite{key1,setA,*keyA1,*keyA2,*keyA3,key2,setB,*keyB1,*keyB2,*keyB3}
```

is essentially equivalent to this:

```
\defbibentryset{setA}{keyA1,keyA2,keyA3}%
\defbibentryset{setB}{keyB1,keyB2,keyB3}%
\cite{key1,setA,key2,setB}
```

The `\mcite` command will work with any style since the `\cite` backend command is controlled by the main citation style as usual. The `mcite` module provides wrappers for the standard commands in §§ 3.7.1 and 3.7.2. See table 7 for an overview. Pre and postnotes as well as starred variants of all commands are also supported. The parameters will be passed to the backend command. For example:

```
\mcite*[pre][post]{setA,*keyA1,*keyA2,*keyA3}
```

will execute:

```
\defbibentryset{setA}{keyA1,keyA2,keyA3}%
\cite*[pre][post]{setA}
```

Note that the `mcite` module is not a compatibility module. It provides commands which are very similar but not identical in syntax and function to `mcite`'s commands. When migrating from `mcite`/`mciteplus` to Biblatex, legacy files must be updated. With `mcite`, the first member of the citation group is also the identifier of the group as a whole. Borrowing an example from the `mcite` manual, this group:

```
\cite{glashow,*salam,*weinberg}
```

Standard Command	mcite-like Command
<code>\cite</code>	<code>\mcite</code>
<code>\Cite</code>	<code>\Mcite</code>
<code>\parencite</code>	<code>\mparencite</code>
<code>\Parencite</code>	<code>\Mparencite</code>
<code>\footcite</code>	<code>\mfootcite</code>
<code>\footcitetext</code>	<code>\mfootcitetext</code>
<code>\textcite</code>	<code>\mtextcite</code>
<code>\Textcite</code>	<code>\Mtextcite</code>
<code>\supercite</code>	<code>\msupercite</code>

Table 6: mcite-like commands

consists of three entries and the entry key of the first one also serves as identifier of the entire group. In contrast to that, a Biblatex entry set is an entity in its own right. Therefore, it requires a unique entry key which is assigned to the set as it is defined:

```
\mcite{set1,*glashow,*salam,*weinberg}
```

Once defined, an entry set is handled like any regular entry in a bib file. When using one of the numeric styles which ship with biblatex and activating its subentry option, it is even possible to refer to set members. See table 7 for some examples. Restating the original definition of the set is redundant, but permissible. In contrast to mciteplus, however, restating a part of the original definition is invalid. Use the entry key of the set instead.

3.8 本地化命令 Localization Commands

The Biblatex package provides translations for key terms such as ‘edition’ or ‘volume’ as well as definitions for language specific features such as the date format and ordinals. These definitions, which are loaded automatically, may be modified or extended in the document preamble or the configuration file with the commands introduced in this section.

```
\DefineBibliographyStrings{<language>}{<definitions>}
```

This command is used to define localization strings. The *<language>* must be a language name known to the babel/polyglossia packages, i. e., one of the identifiers listed in table 2 on page 30. The *<definitions>* are *<key>=<value>* pairs which assign an expression to an identifier:

```
\DefineBibliographyStrings{american}{%
  bibliography = {Bibliography},
  shorthands    = {Abbreviations},
  editor        = {editor},
  editors       = {editors},
}
```

Input	Output	Comment
<code>\mcite{set1,*glashow,*salam,*weinberg}</code>	[1]	Defining and citing the set
<code>\mcite{set1}</code>	[1]	Subsequent citation of the set
<code>\cite{set1}</code>	[1]	Regular <code>\cite</code> works as usual
<code>\mcite{set1,*glashow,*salam,*weinberg}</code>	[1]	Redundant, but permissible
<code>\mcite{glashow}</code>	[1a]	Citing a set member
<code>\cite{weinberg}</code>	[1c]	Regular <code>\cite</code> works as well

Table 7: `mcite`-like syntax (sample output with `style = numeric` and `subentry` option)

A complete list of all keys supported by default is given in § 4.9.2. Note that all expressions should be capitalized as they usually are when used in the middle of a sentence. The Biblatex package will automatically capitalize the first word when required at the beginning of a sentence. Expressions intended for use in headings should be capitalized in a way that is suitable for titling. In contrast to `\DeclareBibliographyStrings`, `\DefineBibliographyStrings` overrides both the full and the abbreviated version of the string. See § 4.9.1 for further details.

`\DefineBibliographyExtras{⟨language⟩}{⟨code⟩}`

This command is used to adapt language specific features such as the date format and ordinals. The `⟨language⟩` must be a language name known to the `babel/polyglossia` packages. The `⟨code⟩`, which may be arbitrary LaTeX code, will usually consist of redefinitions of the formatting commands from § 3.9.2.

`\UndefineBibliographyExtras{⟨language⟩}{⟨code⟩}`

This command is used to restore the original definition of any commands modified with `\DefineBibliographyExtras`. If a redefined command is included in § 3.9.2, there is no need to restore its previous definition since these commands are adapted by all language modules anyway.

`\DefineHyphenationExceptions{⟨language⟩}{⟨text⟩}`

This is a LaTeX frontend to TeX’s `\hyphenation` command which defines hyphenation exceptions. The `⟨language⟩` must be a language name known to the `babel/polyglossia` packages. The `⟨text⟩` is a whitespace-separated list of words. Hyphenation points are marked with a dash:

```
\DefineHyphenationExceptions{american}{%
  hy-phen-ation ex-cep-tion
}
```


`\NewBibliographyString{⟨key⟩}`

This command declares new localization strings, i.e., it initializes a new `⟨key⟩` to be used in the `⟨definitions⟩` of `\DefineBibliographyStrings`. The `⟨key⟩` argument may also be a comma-separated list of key names. The keys listed in § 4.9.2 are defined by default.

3.9 格式命令 **Formatting Commands**

The commands and facilities presented in this section may be used to adapt the format of citations and the bibliography.

3.9.1 一般命令和钩子 **Generic Commands and Hooks**

The commands in this section may be redefined with `\renewcommand` in the document preamble. Note that all commands starting with `\mk...` take one argument. All of these commands are defined in `biblatex.def`.

<code>\bibsetup</code>	Arbitrary code to be executed at the beginning of the bibliography, intended for commands which affect the layout of the bibliography.	
<code>\bibfont</code>	Arbitrary code setting the font used in the bibliography. This is very similar to <code>\bibsetup</code> but intended for switching fonts.	
<code>\citesetup</code>	Arbitrary code to be executed at the beginning of each citation command.	
<code>\newblockpunct</code>	The separator inserted between ‘blocks’ in the sense explained in § 4.7.1. The default definition is controlled by the package option <code>block</code> (see § 3.1.2.1).	
<code>\newunitpunct</code>	The separator inserted between ‘units’ in the sense explained in § 4.7.1. This will usually be a period or a comma plus an interword space. The default definition is a period and a space.	
<code>\finentrypunct</code>	The punctuation printed at the very end of every bibliography entry, usually a period. The default definition is a period.	
<code>\entrysetpunct</code>	The punctuation printed between bibliography subentries of an entry set. The default definition is a semicolon and a space.	Biber only
<code>\bibnamedelima</code>	This delimiter controls the spacing between the elements which make up a name part. It is inserted automatically after the first name element if the element is less than three characters long and before the last element. The default definition is an interword space penalized by the value of the <code>highnamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	Biber only
<code>\bibnamedelimb</code>	This delimiter is inserted between the elements which make up a name part where <code>\bibnamedelima</code> does not apply. The default definition is an interword space penalized by the value of the <code>lownamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	Biber only

<code>\bibnamedelimc</code>	This delimiter controls the spacing between name parts. It is inserted between the name prefix and the last name if <code>useprefix=true</code> . The default definition is an interword space penalized by the value of the <code>highnamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	
<code>\bibnamedelimd</code>	This delimiter is inserted between all name parts where <code>\bibnamedelimc</code> does not apply. The default definition is an interword space penalized by the value of the <code>lownamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	
<code>\bibnamedelimi</code>	This delimiter replaces <code>\bibnamedelima/b</code> after initials. Note that this only applies to initials given as such in the <code>bib</code> file, not to the initials automatically generated by Biblatex which use their own set of delimiters.	Biber only
<code>\bibinitperiod</code>	The punctuation inserted after initials unless <code>\bibinithyphendelim</code> applies. The default definition is a period (<code>\addot</code>). Please refer to § 3.12.4 for further details.	Biber only
<code>\bibinitdelim</code>	The spacing inserted between multiple initials unless <code>\bibinithyphendelim</code> applies. The default definition is an unbreakable interword space. Please refer to § 3.12.4 for further details.	Biber only
<code>\bibinithyphendelim</code>	The punctuation inserted between the initials of hyphenated name parts, replacing <code>\bibinitperiod</code> and <code>\bibinitdelim</code> . The default definition is a period followed by an unbreakable hyphen. Please refer to § 3.12.4 for further details.	Biber only
<code>\bibindexnamedelima</code>	Replaces <code>\bibnamedelima</code> in the index.	
<code>\bibindexnamedelimb</code>	Replaces <code>\bibnamedelimb</code> in the index.	
<code>\bibindexnamedelimc</code>	Replaces <code>\bibnamedelimc</code> in the index.	
<code>\bibindexnamedelimd</code>	Replaces <code>\bibnamedelimd</code> in the index.	
<code>\bibindexnamedelimi</code>	Replaces <code>\bibnamedelimi</code> in the index.	
<code>\bibindexinitperiod</code>	Replaces <code>\bibinitperiod</code> in the index.	
<code>\bibindexinitdelim</code>	Replaces <code>\bibinitdelim</code> in the index.	
<code>\bibindexinithyphendelim</code>	Replaces <code>\bibinithyphendelim</code> in the index.	
<code>\revsdnamepunct</code>	The punctuation to be printed between the first and last name parts when a name is reversed. Here is an example showing a name with the default comma as <code>\revsdnamedelim</code> : Jones, Edward This command should be used with <code>\bibnamedelimd</code> as a reversed-name separator in formatting directives for name lists. Please refer to § 3.12.4 for further details.	

- `\bibnamedash` The dash to be used as a replacement for recurrent authors or editors in the bibliography. The default is an ‘em’ or an ‘en’ dash, depending on the indentation of the list of references.
- `\labelnamepunct` The separator printed after the name used for alphabetizing in the bibliography (author or editor, if the author field is undefined). With the default styles, this separator replaces `\newunitpunct` at this location. The default definition is `\newunitpunct`, i.e., it is not handled differently from regular unit punctuation.
- `\subtitlepunct` The separator printed between the fields title and subtitle, booktitle and booksubtitle, as well as maintitle and mainsubtitle. With the default styles, this separator replaces `\newunitpunct` at this location. The default definition is `\newunitpunct`, i.e., it is not handled differently from regular unit punctuation.
- `\intitlepunct` The separator between the word “in” and the following title in entry types such as `@article`, `@inbook`, `@incollection`, etc. The default definition is a colon plus an interword space (e.g., “Article, in: *Journal*” or “Title, in: *Book*”). Note that this is the separator string, not only the punctuation mark. If you don’t want a colon after “in”, `\intitlepunct` should still insert a space.
- `\bibpagespunct` The separator printed before the pages field. The default is a comma plus an interword space.
- `\bibpagerefspunct` The separator printed before the pageref field. The default is an interword space.
- `\multinamedelim` The delimiter printed between multiple items in a name list like `author` or `editor` if there are more than two names in the list. The default is a comma plus an interword space. See `\finalnamedelim` for an example.²⁰
- `\finalnamedelim` The delimiter printed instead of `\multinamedelim` before the final name in a name list. The default is the localized term ‘and’, separated by interword spaces. Here is an example:

Michel Goossens, Frank Mittelbach `and` Alexander Samarin
Edward Jones `and` Joe Williams

The comma in the first example is the `\multinamedelim` whereas the string ‘and’ in both examples is the `\finalnamedelim`. See also `\finalandcomma` in § 3.9.2.

- `\revsdnamedelim` An extra delimiter printed after the first name in a name list if the first name is reversed. The default is an empty string, i.e., no extra delimiter will be printed. Here is an example showing a name list with a comma as `\revsdnamedelim`:

Jones, Edward, `and` Joe Williams

In this example, the comma after ‘Edward’ is the `\revsdnamedelim` whereas the string ‘and’ is the `\finalnamedelim`, printed in addition to the former.

²⁰Note that `\multinamedelim` is not used at all if there are only two names in the list. In this case, the default styles use the `\finalnamedelim`.

`\andothersdelim` The delimiter printed before the localization string ‘andothers’ if a name list like `author` or `editor` is truncated. The default is an interword space.

`\multilistdelim` The delimiter printed between multiple items in a literal list like `publisher` or `location` if there are more than two items in the list. The default is a comma plus an interword space. See `\multinamedelim` for further explanation.

`\finallistdelim` The delimiter printed instead of `\multilistdelim` before the final item in a literal list. The default is the localized term ‘and’, separated by interword spaces. See `\finalnamedelim` for further explanation.

`\andmoredelim` The delimiter printed before the localization string ‘andmore’ if a literal list like `publisher` or `location` is truncated. The default is an interword space.

`\multicitedelim` The delimiter printed between citations if multiple entry keys are passed to a single citation command. The default is a semicolon plus an interword space.

`\supercitedelim` Similar to `\multicitedelim`, but used by the `\supercite` command only. The default is a comma.

`\compcitedelim` Similar to `\multicitedelim`, but used by certain citation styles when ‘compressing’ multiple citations. The default definition is a comma plus an interword space.

`\textcitedelim` Similar to `\multicitedelim`, but used by `\textcite` and related commands (§ 3.7.2). The default is a comma plus an interword space. The standard styles modify this provisional definition to ensure that the delimiter before the final citation is the localized term ‘and’, separated by interword spaces. See also `\finalandcomma` and `\finalandsemicolon` in § 3.9.2.

`\nametitledelim` The delimiter printed between the author/editor and the title by author-title and some verbose citation styles. The default definition is a comma plus an interword space.

`\nameyear delim` The delimiter printed between the author/editor and the year by author-year citation styles. The default definition is an interword space.

`\labelalphaothers` A string to be appended to the non-numeric portion of the `labelalpha` field (i. e., the field holding the citation label used by alphabetic citation styles) if the number of authors/editors exceeds the `maxalphanames` threshold or the author/editor list was truncated in the `bib` file with the keyword ‘and others’. This will typically be a single character such as a plus sign or an asterisk. The default is a plus sign. This command may also be redefined to an empty string to disable this feature. In any case, it must be redefined in the preamble.

`\sortalphaothers` Similar to `\labelalphaothers` but used in the sorting process. Setting it to a different value is advisable if the latter contains formatting commands, for example: Biber only

```
\renewcommand*{\labelalphaothers}{\textbf{+}}
\renewcommand*{\sortalphaothers}{+}
```

If `\sortalphaothers` is not redefined, it defaults to `\labelalphaothers`.

- `\prenotedelim` The delimiter printed after the `\prenote` argument of a citation command. See § 3.7 for details. The default is an interword space.
- `\postnotedelim` The delimiter printed before the `\postnote` argument of a citation command. See § 3.7 for details. The default is a comma plus an interword space.
- `\mkbibnamelast`{*text*} This command, which takes one argument, is used to format the last name of all authors, editors, translators, etc.
- `\mkbibnamefirst`{*text*} Similar to `\mkbibnamelast`, but intended for the first name.
- `\mkbibnameprefix`{*text*} Similar to `\mkbibnamelast`, but intended for the name prefix.
- `\mkbibnameaffix`{*text*} Similar to `\mkbibnamelast`, but intended for the name affix.
- `\relatedpunct` The separator between the `relatedtype` bibliography localization string and the data from the first related entry. Here is an example with `\relatedpunct` set to a dash:
- A. Smith. Title. 2000, (Orig. pub. as-Origtitle)
- `\relateddelim` The separator between the data of multiple related entries. The default definition is an optional dot plus linebreak. Here is an example where volumes A-E are related entries of the 5 volume main work:
- Donald E. Knuth. Computers & Typesetting. 5 vols. Reading, Mass.: Addison-Wesley, 1984-1986.
 Vol. A: The TEXbook. 1984.
 Vol. B: TEX: The Program. 1986.
 Vol. C: The METAFONTbook. By. 1986.
 Vol. D: METAFONT: The Program. 1986.
 Vol. E: Computer Modern Typefaces. 1986.

3.9.2 语言相关命令 Language-specific Commands

The commands in this section are language specific. When redefining them, you need to wrap the new definition in a `\DeclareBibliographyExtras` command (in an `.ltx` file) or a `\DefineBibliographyExtras` command (user documents), see § 3.8 for details. Note that all commands starting with `\mk...` take one or more arguments.

`\bibrangedash` The language specific dash to be used for ranges of numbers.

`\bibrangessep` Biber only

The language specific separator to be used between multiple ranges.

`\bibdatedash` The language specific dash to be used for date ranges.

`\mkbibdatelong` Takes the names of three field as arguments which correspond to three date components (in the order year/month/day) and uses their values to print the date in the language specific long date format.

`\mkbibdateshort` Similar to `\mkbibdatelong` but using the language specific short date format.

`\finalandcomma` Prints the comma to be inserted before the final ‘and’ in a list, if applicable in the respective language. Here is an example:

```
Michel Goossens, Frank Mittelbach, and Alexander Samarin
```

`\finalandcomma` is the comma before the word ‘and’. See also `\multinamedelim`, `\finalnamedelim`, `\textcitedelim`, and `\revsdsnamedelim` in § 3.9.1.

`\finalandsemicolon` Prints the semicolon to be inserted before the final ‘and’ in a list of lists, if applicable in the respective language. Here is an example:

```
Goossens, Mittelbach, and Samarin; Bertram and Wenworth; and Knuth
```

`\finalandsemicolon` is the semicolon before the word ‘and’. See also `\textcitedelim` in § 3.9.1.

`\mkbibordinal`{*integer*}

This command, which takes an integer as its argument, prints an ordinal number.

`\mkbibmascord`{*integer*}

Similar to `\mkbibordinal`, but prints a masculine ordinal, if applicable in the respective language.

`\mkbibfemord`{*integer*}

Similar to `\mkbibordinal`, but prints a feminine ordinal, if applicable in the respective language.

`\mkbibneutord`{*integer*}

Similar to `\mkbibordinal`, but prints a neuter ordinal, if applicable in the respective language.

`\mkbibordedition`{*integer*}

Similar to `\mkbibordinal`, but intended for use with the term ‘edition’.

`\mkbibordseries{⟨integer⟩}`

Similar to `\mkbibordinal`, but intended for use with the term ‘series’.

3.9.3 尺寸和计数器 Lengths and Counters

The length registers and counters in this section may be changed in the document preamble with `\setlength` and `\setcounter`, respectively.

- `\bibhang` The hanging indentation of the bibliography, if applicable. This length is initialized to `\parindent` at load-time.
- `\biblabelsep` The horizontal space between entries and their corresponding labels in the bibliography. This only applies to bibliography styles which print labels, such as the numeric and alphabetic styles. This length is initialized to twice the value of `\labelsep` at load-time.
- `\bibitemsep` The vertical space between the individual entries in the bibliography. This length is initialized to `\itemsep` at load-time. Note that `\bibitemsep`, `\bibnamesep`, and `\bibinitsep` obey the rules for `\addvspace`, that is, when vertical space introduced by any of these commands immediately follows on from space introduced by another of them, the resulting total space is equal to the largest of them.
- `\bibnamesep` Vertical space to be inserted between two entries in the bibliography whenever an entry starts with a name which is different from the initial name of the previous entry. The default value is zero. Setting this length to a positive value greater than `\bibitemsep` will group the bibliography by author/editor name. Note that `\bibitemsep`, `\bibnamesep`, and `\bibinitsep` obey the rules for `\addvspace`, that is, when vertical space introduced by any of these commands immediately follows on from space introduced by another of them, the resulting total space is equal to the largest of them.
- `\bibinitsep` Vertical space to be inserted between two entries in the bibliography whenever an entry starts with a letter which is different from the initial letter of the previous entry. The default value is zero. Setting this length to a positive value greater than `\bibitemsep` will group the bibliography alphabetically. Note that `\bibitemsep`, `\bibnamesep`, and `\bibinitsep` obey the rules for `\addvspace`, that is, when vertical space introduced by any of these commands immediately follows on from space introduced by another of them, the resulting total space is equal to the largest of them.
- `\bibparsep` The vertical space between paragraphs within an entry in the bibliography. The default value is zero.
- `abbrvpenalty` This counter, which is used by the localization modules, holds the penalty used in short or abbreviated localization strings. For example, a linebreak in expressions such as “et al.” or “ed. by” is unfortunate, but should still be possible to prevent overfull boxes. This counter is initialized to `\hyphenpenalty` at load-time. The idea is making TeX treat the whole expression as if it were a single, hyphenatable word as far as line-breaking is

concerned. If you dislike such linebreaks, use a higher value. If you do not mind them at all, set this counter to zero. If you want to suppress them unconditionally, set it to ‘infinite’ (10 000 or higher).²¹

`highnamepenalty` This counter holds a penalty affecting line-breaking in names. Please refer to §§ 3.12.4 and 3.9.1 for explanation. The counter is initialized to `\hyphenpenalty` at load-time. Use a higher value if you dislike the respective linebreaks. If you do not mind them at all, set this counter to zero. If you prefer the traditional BibTeX behavior (no linebreaks at `highnamepenalty` breakpoints), set it to ‘infinite’ (10 000 or higher).

`lownamepenalty` Similar to `highnamepenalty`. Please refer to §§ 3.12.4 and 3.9.1 for explanation. The counter is initialized to half the `\hyphenpenalty` at load-time. Use a higher value if you dislike the respective linebreaks. If you do not mind them at all, set this counter to zero.

3.9.4 多用途命令 All-purpose Commands

The commands in this section are all-purpose text commands which are generally available, not only in citations and the bibliography.

`\bibellipsis` An ellipsis symbol with brackets: ‘[...]’.

`\noligature` Disables ligatures at this position and adds some space. Use this command to break up standard ligatures like ‘fi’ and ‘fl’. It is similar to the “| shorthand provided by some language modules of the `babel/polyglossia` packages.

`\hyphenate` A conditional hyphen. In contrast to the standard `\-` command, this one allows hyphenation in the rest of the word. It is similar to the “– shorthand provided by some language modules of the `babel/polyglossia` packages.

`\hyphen` An explicit, breakable hyphen intended for compound words. In contrast to a literal ‘–’, this command allows hyphenation in the rest of the word. It is similar to the “= shorthand provided by some language modules of the `babel/polyglossia` packages.

`\nbhyphen` An explicit, non-breakable hyphen intended for compound words. In contrast to a literal ‘–’, this command does not permit line breaks at the hyphen but still allows hyphenation in the rest of the word. It is similar to the “~ shorthand provided by some language modules of the `babel/polyglossia` packages.

`\nohyphenation` A generic switch which suppresses hyphenation locally. Its scope should normally be confined to a group.

²¹The default values assigned to `abbrvpenalty`, `lownamepenalty`, and `highnamepenalty` are deliberately very low to prevent overfull boxes. This implies that you will hardly notice any effect on line-breaking if the text is set justified. If you set these counters to 10 000 to suppress the respective breakpoints, you will notice their effect but you may also be confronted with overfull boxes. Keep in mind that line-breaking in the bibliography is often more difficult than in the body text and that you can not resort to rephrasing a sentence. In some cases it may be preferable to set the entire bibliography `\raggedright` to prevent suboptimal linebreaks. In this case, even the fairly low default penalties will make a visible difference.

`\textnohyphenation{⟨text⟩}`

Similar to `\nohyphenation` but restricted to the `⟨text⟩` argument.

`\mknumalph{⟨integer⟩}`

Takes an integer in the range 1–702 as its argument and converts it to a string as follows: 1=a, ..., 26=z, 27=aa, ..., 702=zz. This is intended for use in formatting directives for the `extrayear` and `extraalpha` fields.

`\mkbibacro{⟨text⟩}`

Generic command which typesets an acronym using the small caps variant of the current font, if available, and as-is otherwise. The acronym should be given in uppercase letters.

`\autocap{⟨character⟩}`

Automatically converts the `⟨character⟩` to its uppercase form if Biblatex’s punctuation tracker would capitalize a localization string at the current location. This command is robust. It is useful for conditional capitalization of certain strings in an entry. Note that the `⟨character⟩` argument is a single character given in lowercase. For example:

`\autocap{s}pecial issue`

will yield ‘Special issue’ or ‘special issue’, as appropriate. If the string to be capitalized starts with an inflected character given in Ascii notation, include the accent command in the `⟨character⟩` argument as follows:

`\autocap{\’e}dition sp\’eciale`

This will yield ‘Édition spéciale’ or ‘édition spéciale’. If the string to be capitalized starts with a command which prints a character, such as `\ae` or `\oe`, simply put the command in the `⟨character⟩` argument:

`\autocap{\oe}uvres`

This will yield ‘Œuvres’ or ‘œuvres’.

3.10 语言相关注意点 Language-specific Notes

The facilities discussed in this section are specific to certain localization modules.

3.10.1 美语 American

The American localization module uses `\uspunctuation` from § 4.7.5 to enable ‘American-style’ punctuation. If this feature is enabled, all trailing commas and periods after `\mkbibquote` will be moved inside the quotes. If you want to disable this feature, use `\stdpunctuation` as follows:

```
\DefineBibliographyExtras{american}{%
  \stdpunctuation
}
```

By default, the ‘American punctuation’ feature is enabled by the `american` localization module only. The above code is only required if you want American localization without American punctuation. Since standard punctuation is the package default, it would be redundant with any other language.

It is highly advisable to always specify `american`, `british`, `australian`, etc. rather than `english` when loading the `babel/polyglossia` packages to avoid any possible confusion. Older versions of the `babel` package used to treat `english` as an alias for `british`; more recent ones treat it as an alias for `american`. The `biblatex` package essentially treats `english` as an alias for `american`, except for the above feature which is only enabled if `american` is requested explicitly.

3.10.2 西班牙语 Spanish

Handling the word ‘and’ is more difficult in Spanish than in the other languages supported by this package because it may be ‘y’ or ‘e’, depending on the initial sound of the following word. Therefore, the Spanish localization module does not use the localization string ‘and’ but a special internal ‘smart and’ command. The behavior of this command is controlled by the `smartand` counter.

`smartand` This counter controls the behavior of the internal ‘smart and’ command. When set to 1, it prints ‘y’ or ‘e’, depending on the context. When set to 2, it always prints ‘y’. When set to 3, it always prints ‘e’. When set to 0, the ‘smart and’ feature is disabled. This counter is initialized to 1 at load-time and may be changed in the preamble. Note that setting this counter to a positive value implies that the Spanish localization module ignores `\finalnamedelim` and `\finallistdelim`.

`\forceE` Use this command in `bib` files if Biblatex gets the ‘and’ before a certain name wrong. As its name suggests, it will enforce ‘e’. This command must be used in a special way to prevent confusing BibTeX. Here is an example:

```
author = {Edward Jones and Eoin Maguire},
author = {Edward Jones and {\forceE{E}}oin Maguire},
```

Note that the initial letter of the respective name component is given as an argument to `\forceE` and that the entire construct is wrapped in an additional pair of curly braces.

`\forceY` Similar to `\forceE` but enforces ‘y’.

3.10.3 希腊语 Greek

The Greek localization module requires UTF-8 support. It will not work with any other encoding. Generally speaking, the Biblatex package is compatible with the `inputenc`

package and with XeLaTeX. The `ucs` package will not work. Since `inputenc`'s standard `utf8` module is missing glyph mappings for Greek, this leaves Greek users with XeLaTeX. Note that you may need to load additional packages which set up Greek fonts. As a rule of thumb, a setup which works for regular Greek documents should also work with Biblatex. However, there is one fundamental limitation. As of this writing, Biblatex has no support for switching scripts. Greek titles in the bibliography should work fine, provided that you use Biber as a backend, but English and other titles in the bibliography may be rendered in Greek letters. If you need multi-script bibliographies, using XeLaTeX is the only sensible choice.

3.10.4 俄语 Russian

Like the Greek localization module, the Russian module also requires UTF-8 support. It will not work with any other encoding.

3.11 用法注意点 Usage Notes

The following sections give a basic overview of the Biblatex package and discuss some typical usage scenarios.

3.11.1 概述 Overview

Using the Biblatex package is slightly different from using traditional BibTeX styles and related packages. Before we get to specific usage scenarios, we will therefore have a look at the structure of a typical document first:

```
\documentclass{...}
\usepackage[...] {biblatex}
\addbibresource{bibfile.bib}
\begin{document}
\cite{...}
...
\printbibliography
\end{document}
```

With traditional BibTeX, the `\bibliography` command serves two purposes. It marks the location of the bibliography and it also specifies the `bib` file(s). The file extension is omitted. With Biblatex, resources are specified in the preamble with `\addbibresource` using the full name with `.bib` suffix. The bibliography is printed using the `\printbibliography` command which may be used multiple times (see § 3.6 for details). The document body may contain any number of citation commands (§ 3.7). Processing this example file requires that a certain procedure be followed. Suppose our example file is called `example.tex` and our bibliographic data is in `bibfile.bib`. The procedure, then, is as follows:

3.11.1.1 Biber

1. Run `latex` on `example.tex`. If the file contains any citations, Biblatex will request the respective data from Biber by writing commands to the auxiliary file `example.bcf`.
2. Run `biber` on `example.bcf`. Biber will retrieve the data from `bibfile.bib` and write it to the auxiliary file `example.bbl` in a format which can be processed by Biblatex.
3. Run `latex` on `example.tex`. Biblatex will read the data from `example.bbl` and print all citations as well as the bibliography.

3.11.1.2 BibTeX

1. Run `latex` on `example.tex`. If the file contains any citations, Biblatex will request the respective data from BibTeX by writing commands to the auxiliary file `example.aux`.
2. Run `bibtex` on `example.aux`. BibTeX will retrieve the data from `bibfile.bib` and write it to the auxiliary file `example.bbl` in a format which can be processed by Biblatex.
3. Run `latex` on `example.tex`. Biblatex will read the data from `example.bbl` and print all citations as well as the bibliography.

Whenever a reference to a work which has not been cited before is added, this procedure must be repeated. This is also the case if the last reference to a work which has been cited before is removed because some citation labels may change in this case. In contrast to traditional BibTeX, there is normally no need to run `latex` twice after running the backend as far as the handling of bibliographic data is concerned.²²

Note that when using BibTeX as the backend this only applies to the most basic case. Using the `xref` field or the `entryset` field may require an additional LaTeX/BibTeX/LaTeX cycle. Some other facilities provided by Biblatex may also require an additional `latex` run to get certain references and the page tracking right. In this case, the usual warning messages such as “There were undefined references” and “Label(s) may have changed. Rerun to get cross-references right” will be printed.

BibTeX only

3.11.2 辅助文件 Auxiliary Files

3.11.2.1 Biber The Biblatex package uses one auxiliary `bcf` file only. Even if there are citation commands in a file included via `\include`, you only need to run Biber on the main `bcf` file. All information Biber needs is in the `bcf` file, including information about all refsections if using multiple `refsection` environments (see § 3.11.3).

²²That is, unless the `defernumbers` package option is enabled. See § 4.1

3.11.2.2 BibTeX By default, the Biblatex package uses the main aux file only. Even if there are citation commands in a file included via `\include`, which has its own aux file, you only need to run BibTeX on the main aux file. If you are using `refsection` environments in a document (see § 3.11.3) Biblatex will create one additional aux file for every `refsection` environment. In this case, you also need to run `bibtex` on each additional aux file. The name of the additional aux files is the base name of the main input file with the string `-blx` and a running number appended at the end. The Biblatex package issues a warning listing the files which require an additional BibTeX run. With the basic example presented in § 3.11.1, it would issue the following warning:

```
Package biblatex Warning: Please (re)run BibTeX on the file(s):
(biblatex)                example.aux
(biblatex)                and rerun LaTeX afterwards.
```

If the input file contained three `refsection` environments, the warning would read as follows:

```
Package biblatex Warning: Please (re)run BibTeX on the file(s):
(biblatex)                example1-blx.aux
(biblatex)                example2-blx.aux
(biblatex)                example3-blx.aux
(biblatex)                and rerun LaTeX afterwards.
```

Apart from these aux files, Biblatex uses an additional `bib` file with the same suffix to pass certain control parameters to BibTeX. In the example above, this file would be named `example-blx.bib`. In the event of a file name conflict, you can change the suffix by redefining the macro `\blxauxsuffix` in the document preamble. When using Biber, Biblatex writes a control file named `example.bcf` and ignores `\blxauxsuffix`. There is also no auxiliary `bib` file in this case.

Note that Biblatex will not overwrite any files it did not create. All auxiliary files created automatically by this package start with a special signature line. Before overwriting a file (excluding the main aux file, which is managed by LaTeX), Biblatex inspects the first line of the file to make sure that there is no file name conflict. If the file in question is missing the signature line, Biblatex will immediately issue an error message and abort before opening the output stream. In this case you should delete any spurious files accidentally left in the working directory. If the error persists, there may be a file name conflict with a file found in one of the TeX installation trees. Since the installation trees usually do not contain any aux files and the string `-blx` is fairly exotic in the name of a `bib` file, this is rather unlikely but theoretically possible. If you find out that this is indeed the case, you should redefine `\blxauxsuffix` permanently in the Biblatex configuration file, `biblatex.cfg`.

3.11.3 多个参考文献表 Multiple Bibliographies

In a collection of articles by different authors, such as a conference proceedings volume for example, it is very common to have one bibliography for each article rather than a

global one for the entire book. In the example below, each article would be presented as a separate `\chapter` with its own bibliography.

Note that with the BibTeX backend, Biblatex creates one additional aux file for every `refsection` environment. These files have to be processed by BibTeX as well, see § 3.11.2 for details. BibTeX only

```
\documentclass{...}
\usepackage{biblatex}
\addbibresource{...}
\begin{document}
\chapter{...}
\begin{refsection}
...
\printbibliography[heading=subbibliography]
\end{refsection}
\chapter{...}
\begin{refsection}
...
\printbibliography[heading=subbibliography]
\end{refsection}
\end{document}
```

If `\printbibliography` is used inside a `refsection` environment, it automatically restricts the scope of the list of references to the enclosing `refsection` environment. For a cumulative bibliography which is subdivided by chapter but printed at the end of the book, use the `section` option of `\printbibliography` to select a reference section, as shown in the next example.

```
\documentclass{...}
\usepackage{biblatex}
\defbibheading{subbibliography}{%
  \section*{References for Chapter \ref{refsection:\therefsection}}}
\addbibresource{...}
\begin{document}
\chapter{...}
\begin{refsection}
...
\end{refsection}
\chapter{...}
\begin{refsection}
...
\end{refsection}
\printbibheading
```

```

\printbibliography[section=1,heading=subbibliography]
\printbibliography[section=2,heading=subbibliography]
\end{document}

```

Note the definition of the bibliography heading in the above example. This is the definition taking care of the subheadings in the bibliography. The main heading is generated with a plain `\chapter` command in this case. The Biblatex package automatically sets a label at the beginning of every `refsection` environment, using the standard `\label` command. The identifier used is the string `refsection:` followed by the number of the respective `refsection` environment. The number of the current section is accessible via the `refsection` counter. When using the `section` option of `\printbibliography`, this counter is also set locally. This means that you may use the counter in heading definitions to print subheadings like “References for Chapter 3”, as shown above. You could also use the title of the respective chapter as a subheading by loading the `nameref` package and using `\nameref` instead of `\ref`:

```

\usepackage{nameref}
\defbibheading{subbibliography}{%
  \section*{\nameref{refsection:\therefsection}}
}

```

Since giving one `\printbibliography` command for each part of a subdivided bibliography is tedious, Biblatex provides a shorthand. The `\bibbysection` command automatically loops over all reference sections. This is equivalent to giving one `\printbibliography` command for every section but has the additional benefit of automatically skipping sections without references. In the example above, the bibliography would then be generated as follows:

```

\printbibheading
\bibbysection[heading=subbibliography]

```

When using a format with one cumulative bibliography subdivided by chapter (or any other document division) it may be more appropriate to use `refsegment` rather than `refsection` environments. The difference is that the `refsection` environment generates labels local to the environment while `refsegment` does not affect the generation of labels, hence they will be unique across the entire document. Note that when using BibTeX as the backend, `refsegment` environments do not require additional aux files. The next example could also be given in § 3.11.4 because, visually, it creates one global bibliography subdivided into multiple segments.

```

\documentclass{...}
\usepackage{biblatex}
\defbibheading{subbibliography}{%
  \section*{References for Chapter \ref{refsegment:\therefsection\therefsegment}}
\addbibresource{...}

```

```

\begin{document}
\chapter{...}
\begin{refsegment}
...
\end{refsegment}
\chapter{...}
\begin{refsegment}
...
\end{refsegment}
\printbibheading
\printbibliography[segment=1,heading=subbibliography]
\printbibliography[segment=2,heading=subbibliography]
\end{document}

```

The use of `refsegment` is similar to `refsection` and there is also a corresponding `segment` option for `\printbibliography`. The Biblatex package automatically sets a label at the beginning of every `refsegment` environment using the string `refsegment:` followed by the number of the respective `refsegment` environment as an identifier. There is a matching `refsegment` counter which may be used in heading definitions, as shown above. As with reference sections, there is also a shorthand command which automatically loops over all reference segments:

```

\printbibheading
\bibbysegment[heading=subbibliography]

```

This is equivalent to giving one `\printbibliography` command for every segment in the current `refsection`.

3.11.4 子参考文献表 Subdivided Bibliographies

It is very common to subdivide a bibliography by certain criteria. For example, you may want to list printed and online resources separately or divide a bibliography into primary and secondary sources. The former case is straightforward because you can use the entry type as a criterion for the `type` and `notttype` filters of `\printbibliography`. The next example also demonstrates how to generate matching subheadings for the two parts of the bibliography.

```

\documentclass{...}
\usepackage{biblatex}
\addbibresource{...}
\begin{document}
...
\printbibheading
\printbibliography[notttype=online,heading=subbibliography,

```



```

        title={Printed Sources}]
\printbibliography[type=online,heading=subbibliography,
        title={Online Sources}]

\end{document}

```

You may also use more than two subdivisions:

```

\printbibliography[type=article,...]
\printbibliography[type=book,...]
\printbibliography[notttype=article,notttype=book,...]

```

It is even possible to give a chain of different types of filters:

```

\printbibliography[section=2,type=book,keyword=abc,notkeyword=xyz]

```

This would print all works cited in reference section 2 whose entry type is @book and whose keywords field includes the keyword ‘abc’ but not ‘xyz’. When using bibliography filters in conjunction with a numeric style, see § 3.12.5. If you need complex filters with conditional expressions, use the `filter` option in conjunction with a custom filter defined with `\defbibfilter`. See § 3.6.10 for details on custom filters.

```

\documentclass{...}
\usepackage{biblatex}
\addbibresource{...}
\begin{document}
...
\printbibheading
\printbibliography[keyword=primary,heading=subbibliography,%
        title={Primary Sources}]
\printbibliography[keyword=secondary,heading=subbibliography,%
        title={Secondary Sources}]
\end{document}

```

Dividing a bibliography into primary and secondary sources is possible with a keyword filter, as shown in the above example. In this case, with only two subdivisions, it would be sufficient to use one keyword as filter criterion:

```

\printbibliography[keyword=primary,...]
\printbibliography[notkeyword=primary,...]

```

Since Biblatex has no way of knowing if an item in the bibliography is considered to be primary or secondary literature, we need to supply the bibliography filter with the required data by adding a `keywords` field to each entry in the `bib` file. These keywords may then be used as targets for the `keyword` and `notkeyword` filters, as

shown above. It may be a good idea to add such keywords right away while building a bib file.

```
@Book{key,  
  keywords      = {primary,some,other,keywords},  
  ...
```

An alternative way of subdividing the list of references are bibliography categories. They differ from the keywords-based approach shown in the example above in that they work on the document level and do not require any changes to the bib file.

```
\documentclass{...}  
\usepackage{biblatex}  
\DeclareBibliographyCategory{primary}  
\DeclareBibliographyCategory{secondary}  
\addtocategory{primary}{key1,key3,key6}  
\addtocategory{secondary}{key2,key4,key5}  
\addbibresource{...}  
\begin{document}  
...  
\printbibheading  
\printbibliography[category=primary,heading=subbibliography,%  
                    title={Primary Sources}]  
\printbibliography[category=secondary,heading=subbibliography,%  
                    title={Secondary Sources}]  
\end{document}
```

In this case it would also be sufficient to use one category only:

```
\printbibliography[category=primary,...]  
\printbibliography[notcategory=primary,...]
```

It is still a good idea to declare all categories used in the bibliography explicitly because there is a `\bibbycategory` command which automatically loops over all categories. This is equivalent to giving one `\printbibliography` command for every category, in the order in which they were declared.

```
\documentclass{...}  
\usepackage{biblatex}  
\DeclareBibliographyCategory{primary}  
\DeclareBibliographyCategory{secondary}  
\addtocategory{primary}{key1,key3,key6}  
\addtocategory{secondary}{key2,key4,key5}  
\defbibheading{primary}{\section*{Primary Sources}}
```

```

\defbibheading{secondary}{\section*{Secondary Sources}}
\addbibresource{...}
\begin{document}
...
\printbibheading
\bibbcategory
\end{document}

```

The handling of the headings is different from `\bibbysection` and `\bibbysegment` in this case. `\bibbcategory` uses the name of the current category as a heading name. This is equivalent to passing `heading=\langle category \rangle` to `\printbibliography` and implies that you need to provide a matching heading for every category.

3.11.5 条目集 Entry Sets

An entry set is a group of entries which are cited as a single reference and listed as a single item in the bibliography. The individual entries in the set are separated by `\entrysetpunct` (§ 4.10.1). The Biblatex package supports two types of entry sets. Static entry sets are defined in the `bib` file like any other entry. Dynamic entry sets are defined with `\defbibentryset` (§ 3.6.12) on a per-document/per-refsection basis in the document preamble or the document body. This section deals with the definition of entry sets; style authors should also see § 4.11.1 for further information.

3.11.5.1 Static entry sets Static entry sets are defined in the `bib` file like any other entry. When using Biber as the backend, defining an entry set is as simple as adding an entry of type `@set`. The entry has an `entryset` field defining the members of the set as a separated list of entry keys: Biber only

```

@Set{set1,
  entryset = {key1, key2, key3},
}

```

Entries may be part of a set in one document/refsection and stand-alone references in another one, depending on the presence of the `@set` entry. If the `@set` entry is cited, the set members are grouped automatically. If not, they will work like any regular entry.

When using BibTeX as the backend, which has no native support for entry sets, setting up entry sets involves more work. BibTeX requires `entryset` and `crossref` fields to be used in a special way. The members of the set are given in the `entryset` field of the `@set` entry. The `@set` entry also requires a `crossref` field which points to the first key in the `entryset` field. In addition to that, all members of the set require `entryset` fields which are reverse pointers to the entry key of the `@set` head entry:

```

@Set{set1,
  entryset = {key1, key2, key3},
  crossref = {key1},
}
@Article{key1,
  entryset = {set1},
  author   = {...},
  title    = {...},
  ...
}
@InCollection{key2,
  entryset = {set1},
  author   = {...},
  title    = {...},
  ...
}
@Article{key3,
  entryset = {set1},
  author   = {...},
  title    = {...},
  ...
}

```

Note that citing any set member will automatically load the entire set with BibTeX. If you want to refer to an item as part of a set in one document/refsection and as a stand-alone reference in another one, you need two distinct entries with BibTeX.

3.11.5.2 Dynamic entry sets Dynamic entry sets are set up and work much like static ones. The main difference is that they are defined in the document preamble or on the fly in the document body using the `\defbibentryset` command from § 3.6.12:

Biber only

```
\defbibentryset{set1}{key1, key2, key3}
```

Dynamic entry sets in the document body are local to the enclosing `refsection` environment, if any. Otherwise, they are assigned to reference section 0. Those defined in the preamble are assigned to reference section 0. Note that dynamic entry sets require Biber. They will not work with any other backend.

3.11.6 数据容器 Data Containers

Biber only

The `@xdata` entry type serves as a data container holding one or more fields. These fields may be inherited by other entries using the `xdata` field. `@xdata` entries may not be cited or added to the bibliography, they only serve as a data source for other

entries. This data inheritance mechanism is useful for fixed field combinations such as publisher/location and for other frequently used data:

```
@XData{hup,  
  publisher = {Harvard University Press},  
  location  = {Cambridge, Mass.},  
}  
@Book{...,  
  author    = {...},  
  title     = {...},  
  date      = {...},  
  xdata     = {hup},  
}
```

Using a separated list of keys in its xdata field, an entry may inherit data from several @xdata entries. Cascading @xdata entries are supported as well, i. e., an @xdata entry may reference one or more other @xdata entries:

```
@XData{macmillan:name,  
  publisher = {Macmillan},  
}  
@XData{macmillan:place,  
  location  = {New York and London},  
}  
@XData{macmillan,  
  xdata     = {macmillan:name,macmillan:place},  
}  
@Book{...,  
  author    = {...},  
  title     = {...},  
  date      = {...},  
  xdata     = {macmillan},  
}
```

See also §§ 2.1.1 and 2.2.3.

3.11.7 电子出版信息 Electronic Publishing Information

The Biblatex package provides three fields for electronic publishing information: eprint, eprinttype, and eprintclass. The eprint field is a verbatim field similar to doi which holds the identifier of the item. The eprinttype field holds the resource name, i. e., the name of the site or electronic archive. The optional eprintclass field is intended for additional information specific to the resource indicated by the eprinttype field. This could be a section, a path, classification information, etc. If the eprinttype field is available, the standard styles will use it as

a literal label. In the following example, they would print “Resource: identifier” rather than the generic “eprint: identifier”:

```
eprint      = {identifier},  
eprinttype = {Resource},
```

The standard styles feature dedicated support for a few online archives. For arXiv references, put the identifier in the `eprint` field and the string `arxiv` in the `eprinttype` field:

```
eprint      = {math/0307200v3},  
eprinttype = {arxiv},
```

For papers which use the new identifier scheme (April 2007 and later) add the primary classification in the `eprintclass` field:

```
eprint      = {1008.2849v1},  
eprinttype = {arxiv},  
eprintclass = {cs.DS},
```

There are two aliases which ease the integration of arXiv entries. `archiveprefix` is treated as an alias for `eprinttype`; `primaryclass` is an alias for `eprintclass`. If hyperlinks are enabled, the `eprint` identifier will be transformed into a link to `arxiv.org`. See the package option `arxiv` in § 3.1.2.1 for further details.

For JSTOR references, put the stable JSTOR number in the `eprint` field and the string `jstor` in the `eprinttype` field:

```
eprint      = {number},  
eprinttype = {jstor},
```

When using JSTOR’s export feature to export citations in BibTeX format, JSTOR uses the `url` field by default (where the $\langle number \rangle$ is a unique and stable identifier):

```
url = {http://www.jstor.org/stable/number},
```

While this will work as expected, full URLs tend to clutter the bibliography. With the `eprint` fields, the standard styles will use the more readable “JSTOR: $\langle number \rangle$ ” format which also supports hyperlinks. The $\langle number \rangle$ becomes a clickable link if `hyperref` support is enabled.

For PubMed references, put the stable PubMed identifier in the `eprint` field and the string `pubmed` in the `eprinttype` field. This means that:

```
url = {http://www.ncbi.nlm.nih.gov/pubmed/pmid},
```

becomes:

```
eprint      = {pmid},  
eprinttype = {pubmed},
```

and the standard styles will print “PMID: $\langle pmid \rangle$ ” instead of the lengthy URL. If hyperref support is enabled, the $\langle pmid \rangle$ will be a clickable link to PubMed.

For handles (HDLs), put the handle in the `eprint` field and the string `hdl` in the `eprinttype` field:

```
eprint      = {handle},  
eprinttype = {hdl},
```

For Google Books references, put Google’s identifier in the `eprint` field and the string `googlebooks` in the `eprinttype` field. This means that, for example:

```
url = {http://books.google.com/books?id=XXu4AkRVBBoC},
```

would become:

```
eprint      = {XXu4AkRVBBoC},  
eprinttype = {googlebooks},
```

and the standard styles would print “Google Books: XXu4AkRVBBoC” instead of the full URL. If hyperref support is enabled, the identifier will be a clickable link to Google Books.²³

Note that `eprint` is a verbatim field. Always give the identifier in its unmodified form. For example, there is no need to replace `_` with `_`. Also see § 4.11.2 on how to add dedicated support for other `eprint` resources.

3.11.8 额外的摘要和注释 External Abstracts and Annotations

Styles which print the fields `abstract` and/or `annotation` may support an alternative way of adding abstracts or annotations to the bibliography. Instead of including the text in the `bib` file, it may also be stored in an external LaTeX file. For example, instead of saying

```
@Article{key1,  
  ...  
  abstract      = {This is an abstract of entry 'key1'.}  
}
```

in the `bib` file, you may create a file named `bibabstract-key1.tex` and put the abstract in this file:

```
This is an abstract of entry 'key1'.  
\endinput
```

²³Note that the Google Books ID seems to be a bit of an ‘internal’ value. As of this writing, there does not seem to be any way to search for an ID on Google Books. You may prefer to use the `url` in this case.

The name of the external file must be the entry key prefixed with `bibabstract-` or `bibannotation-`, respectively. You can change these prefixes by redefining `\bibabstractprefix` and `\bibannotationprefix`. Note that this feature needs to be enabled explicitly by setting the package option `loadfiles` from § 3.1.2.1. The option is disabled by default for performance reasons. Also note that any `abstract` and `annotation` fields in the `bib` file take precedence over the external files. Using external files is strongly recommended if you have long abstracts or a lot of annotations since this may increase memory requirements significantly. It is also more convenient to edit the text in a dedicated LaTeX file. Style authors should see § 4.11.3 for further information.

3.12 提示与警告 Hints and Caveats

This section provides additional usage hints and addresses some common problems and potential misconceptions.

3.12.1 与KOMA-Script 类共用的方法 Usage with KOMA-Script Classes

When using Biblatex in conjunction with one of the `scrbook`, `scrreprt`, or `scrartcl` classes, the headings `bibliography` and `biblist` from § 3.6.8 are responsive to the bibliography-related options of these classes.²⁴ You can override the default headings by using the `heading` option of `\printbibliography`, `\printbibheading` and `\printbiblist`. See §§ 3.6.2, 3.6.4, 3.6.8 for details. All default headings are adapted at load-time such that they blend with the behavior of these classes. If one of the above classes is detected, Biblatex will also provide the following additional tests which may be useful in custom heading definitions:

```
\ifkomabibtotoc{<true>}{<false>}
```

Expands to `<true>` if the class would add the bibliography to the table of contents, and to `<false>` otherwise.

```
\ifkomabibtotocnumbered{<true>}{<false>}
```

Expands to `<true>` if the class would add the bibliography to the table of contents as a numbered section, and to `<false>` otherwise. If this test yields `<true>`, `\ifkomabibtotoc` will always yield `<true>` as well, but not vice versa.

3.12.2 与 Memoir 类共用的方法 Usage with the Memoir Class

When using Biblatex with the `memoir` class, most class facilities for adapting the bibliography have no effect. Use the corresponding facilities of this package instead (§§ 3.6.2,

²⁴This applies to the traditional syntax of these options (`bibtotoc` and `bibtotocnumbered`) as well as to the `<key>=<value>` syntax introduced in KOMA-Script 3.x, i.e., to `bibliography=nottotoc`, `bibliography=totoc`, and `bibliography=totocnumbered`. The global `toc=bibliography` and `toc=bibliographynumbered` options as well as their aliases are detected as well. In any case, the options must be set globally in the optional argument to `\documentclass`.

3.6.8, 3.6.9). Instead of redefining memoir’s `\bibsection`, use the heading option of `\printbibliography` and `\defbibheading` (§§ 3.6.2 and 3.6.8). Instead of `\prebibhook` and `\postbibhook`, use the `prenote` and `postnote` options of `\printbibliography` and `\defbibnote` (§§ 3.6.2 and 3.6.9). All default headings are adapted at load-time such that they blend well with the default layout of this class. The default headings `bibliography` and `biblist` (§ 3.6.8) are also responsive to memoir’s `\bibintoc` and `\nobibintoc` switches. The length register `\bibitemsep` is used by Biblalex in a way similar to memoir (§ 3.9.3). This section also introduces some additional length registers which correspond to memoir’s `\biblistextra`. Lastly, `\setbiblabel` does not map to a single facility of the Biblalex package since the style of all labels in the bibliography is controlled by the bibliography style. See § 4.2.2 in the author section of this manual for details. If the memoir class is detected, Biblalex will also provide the following additional test which may be useful in custom heading definitions:

```
\ifmemoirbibintoc{<true>}{<false>}
```

Expands to `<true>` or `<false>`, depending on memoir’s `\bibintoc` and `\nobibintoc` switches. This is a LaTeX frontend to memoir’s `\ifnobibintoc` test. Note that the logic of the test is reversed.

3.12.3 标注中的页码 Page Numbers in Citations

If the `<postnote>` argument to a citation command is a page number or page range, Biblalex will automatically prefix it with ‘p.’ or ‘pp.’ by default. This works reliably in typical cases, but sometimes manual intervention may be required. In this case, it is important to understand how this argument is handled in detail. First, Biblalex checks if the postnote is an Arabic or Roman numeral (case insensitive). If this test succeeds, the postnote is considered as a single page or other number which will be prefixed with ‘p.’ or some other string which depends on the `pagination` field (see § 2.3.10). If it fails, a second test is performed to find out if the postnote is a range or a list of Arabic or Roman numerals. If this test succeeds, the postnote will be prefixed with ‘pp.’ or some other string in the plural form. If it fails as well, the postnote is printed as is. Note that both tests expand the `<postnote>`. All commands used in this argument must therefore be robust or prefixed with `\protect`. Here are a few examples of `<postnote>` arguments which will be correctly recognized as a single number, a range of numbers, or a list of numbers, respectively:

```
\cite[25]{key}
\cite[vii]{key}
\cite[XIV]{key}
\cite[34--38]{key}
\cite[iv--x]{key}
\cite[185/86]{key}
\cite[XI \& XV]{key}
```

```
\cite[3, 5, 7]{key}
\cite[vii--x; 5, 7]{key}
```

In some other cases, however, the tests may get it wrong and you need to resort to the auxiliary commands `\pno`, `\ppno`, and `\nopp` from § 3.7.8. For example, suppose a work is cited by a special pagination scheme consisting of numbers and letters. In this scheme, the string ‘27a’ would mean ‘page 27, part a’. Since this string does not look like a number or a range to Biblatex, you need to force the prefix for a single number manually:

```
\cite[\pno~27a]{key}
```

There is also a `\ppno` command which forces a range prefix as well as a `\nopp` command which suppresses all prefixes:

```
\cite[\ppno~27a--28c]{key}
\cite[\nopp 25]{key}
```

These commands may be used anywhere in the *⟨postnote⟩* argument. They may also be used multiple times. For example, when citing by volume and page number, you may want to suppress the prefix at the beginning of the postnote and add it in the middle of the string:

```
\cite[VII, \pno~5]{key}
\cite[VII, \pno~3, \ppno~40--45]{key}
\cite[see][\ppno~37--46, in particular \pno~40]{key}
```

There are also two auxiliary command for suffixes like ‘the following page(s)’. Instead of inserting such suffixes literally (which would require `\ppno` to force a prefix):

```
\cite[\ppno~27~sq.]{key}
\cite[\ppno~55~sq.]{key}
```

use the auxiliary commands `\psq` and `\psqq`. Note that there is no space between the number and the command. This space will be inserted automatically and may be modified by redefining the macro `\sqspace`.

```
\cite[27\psq]{key}
\cite[55\psqq]{key}
```

Since the postnote is printed without any prefix if it includes any character which is not an Arabic or Roman numeral, you may also type the prefix manually:

```
\cite[p.~5]{key}
```

It is possible to suppress the prefix on a per-entry basis by setting the `pagination` field of an entry to ‘none’, see § 2.3.10 for details. If you do not want any prefixes at all or prefer to type them manually, you can also disable the entire mechanism in the document preamble or the configuration file as follows:

```
\DeclareFieldFormat{postnote}{#1}
```

The `<postnote>` argument is handled as a field and the formatting of this field is controlled by a field formatting directive which may be freely redefined. The above definition will simply print the postnote as is. See §§ 4.3.2 and 4.4.2 in the author guide for further details.

3.12.4 姓名组成部分及其间距 Name Parts and Name Spacing

The Biblatex package gives users and style authors very fine-grained control of name spacing and the line-breaking behavior of names, especially when they are using Biber as the backend. The commands discussed in the following are documented in §§ 3.9.1 and 4.10.1. This section is meant to give an overview of how they are put together. A note on terminology: a name *part* is a basic part of the name, for example the first or the last name. Each part of a name may be a single name or it may be composed of multiple names. For example, the name part ‘first name’ may be composed of a first and a middle name. The latter are referred to as name *elements* in this section. Let’s consider a simple name first: “John Edward Doe”. This name is composed of the following parts:

```
First   John Edward
Prefix  —
Last    Doe
Suffix  —
```

The spacing, punctuation and line-breaking behavior of names is controlled by six macros:

<code>a=\bibnamedelima</code>	Inserted by the backend after the first element of every name part if that element is less than three characters long and before the last element of every name part.
<code>b=\bibnamedelimb</code>	Inserted by the backend between all elements of a name part where <code>\bibnamedelima</code> does not apply.
<code>c=\bibnamedelimc</code>	Inserted by a formatting directive between the name prefix and the last name if <code>useprefix=true</code> . If <code>useprefix=false</code> , <code>\bibnamedelimd</code> is used instead.
<code>d=\bibnamedelimd</code>	Inserted by a formatting directive between name parts where <code>\bibnamedelimc</code> does not apply.
<code>i=\bibnamedelimi</code>	Replaces <code>\bibnamedelima/b</code> after initials
<code>p=\revsdsnamepunct</code>	Inserted by a formatting directive after the last name when the name parts are reversed.

This is how the delimiters are employed:

```
John|_a Edward|_d Doe
Doe|_p|_d John|_a Edward
```

Initials in the bib file get a special delimiter:

J._i Edward_d Doe

Let's consider a more complex name: "Charles-Jean Étienne Gustave Nicolas de La Vallée Poussin". This name is composed of the following parts:

First Charles-Jean Étienne Gustave Nicolas

Prefix de

Last La Vallée Poussin

Suffix —

The delimiters:

Charles-Jean_b Étienne_b Gustave_a Nicolas_d de_c La_a Vallée_a Poussin

Note that `\bibnamedelima/b/i` are inserted by the backend. The backend processes the name parts and takes care of the delimiters between the elements that make up a name part, processing each part individually. In contrast to that, the delimiters between the parts of the complete name (`\bibnamedelimc/d`) are added by name formatting directives at a later point in the processing chain. The spacing and punctuation of initials is also handled by the backend and may be customized by redefining the following three macros:

<code>a=\bibinitperiod</code>	Inserted by the backend after initials.
<code>b=\bibinitdelim</code>	Inserted by the backend between multiple initials.
<code>c=\bibinithyphendelim</code>	Inserted by the backend between the initials of hyphenated name parts, replacing <code>\bibinitperiod</code> and <code>\bibinitdelim</code> .

This is how they are employed:

J._a E_b Doe

K_c-H_a Mustermann

3.12.5 参考文献筛选器和标注标签 Bibliography Filters and Citation Labels

The citation labels generated by this package are assigned to the full list of references before it is split up by any bibliography filters. They are guaranteed to be unique across the entire document (or a `refsection` environment), no matter how many bibliography filters you are using. When using a numeric citation scheme, however, this will most likely lead to discontinuous numbering in split bibliographies. Use the `defernumbers` package option to avoid this problem. If this option is enabled, numeric labels are assigned the first time an entry is printed in any bibliography.

3.12.6 参考文献标题中的活动字符 Active Characters in Bibliography Headings

Packages using active characters, such as `babel`, `polyglossia`, `csquotes`, or `underscore`, usually do not make them active until the beginning of the document body to avoid interference with other packages. A typical example of such an active character is the Ascii quote `"`, which is used by various language modules of the `babel/polyglossia` packages. If shorthands such as `"<` and `"a` are used in the argument to `\defbibheading` and the headings are defined in the document preamble, the non-active form of the characters is saved in the heading definition. When the heading is typeset, they do not function as a command but are simply printed literally. The most straightforward solution consists in moving `\defbibheading` after `\begin{document}`. Alternatively, you may use `babel`'s `\shorthandon` and `\shorthandoff` commands to temporarily make the shorthands active in the preamble. The above also applies to bibliography notes and the `\defbibnote` command.

3.12.7 在参考文献分节和分部中的编组 Grouping in Reference Sections and Segments

All LaTeX environments enclosed in `\begin` and `\end` form a group. This may have undesirable side effects if the environment contains anything that does not expect to be used within a group. This issue is not specific to `refsection` and `refsegment` environments, but it obviously applies to them as well. Since these environments will usually enclose much larger portions of the document than a typical `itemize` or similar environment, they are simply more likely to trigger problems related to grouping. If you observe any malfunctions after adding `refsection` environments to a document (for example, if anything seems to be ‘trapped’ inside the environment), try the following syntax instead:

```
\chapter{...}  
\refsection  
...  
\endrefsection
```

This will not form a group, but otherwise works as usual. As far as Biblatex is concerned, it does not matter which syntax you use. The alternative syntax is also supported by the `refsegment` environment. Note that the commands `\newrefsection` and `\newrefsegment` do not form a group. See §§ 3.6.5 and 3.6.6 for details.

4 样式作者指南

手册的这一部分介绍 Biblatex 包的作者接口。该指南包括了编写参考文献表和引用样式或本地化模型所需知晓的所有内容。在阅读本部分前最好先阅读用户指南。

4.1 概述 Overview

在讨论 Biblatex 提供的命令和工具之前，我们将首先介绍一些基本概念。Biblatex 包以一种特殊的方式使用辅助文件。最需要注意的是使用了bbl文件，而且当使用 BibTeX 后端时，只有一个bst文件实现用来结构化的数据接口，而不是输出可打印的数据。使用 LaTeX 的标准参考文献工具，一个文档通常包括任意数量的文献引用命令，再加上常放在文档最后的\bibliographystyle和\bibliography命令。文献引用命令在文档中的位置是任意的，而\bibliographystyle和\bibliography命令则标记了打印参考文献表的位置，比如：

```
\documentclass{...}
\begin{document}
\cite{...}
...
\bibliographystyle{...}
\bibliography{...}
\end{document}
```

处理这些文件需要如下的一定的流程，如下：

1. 运行 latex: 第一次运行latex，\bibstyle和\bibdata命令，以及所有引用的命令\citation写入到aux文件中。在这个时候，各引文标注²⁵是未定义的，因为 LaTeX 等待 BibTeX 提供需要的数据，当然参考文献表也不存在。
2. 运行 bibtex:BibTeX 在bbl文件中写入一个thebibliography环境，提供aux文件中\citation命令所需求所有条目，这些条目数据来自bib文件。
3. 运行 latex，第二次运行 latex，thebibliography环境中的\bibitem命令在aux文件中为每个参考文献条目写入\bibcite命令。这些\bibcite命令定义的标签将用于\cite命令。然而，各引文标注仍然未定义，因为这些标签在最后一次运行 latex 前仍未得到。
4. 运行 latex: 第三次运行，这时引用的标签已经定义因为在导言区最后读入了aux文件。这样所有的标注可以正确打印出来。

注意到所有的参考文献数据都以最终格式 (指最后打印出的格式) 写入bbl文件。该文件的读取和处理如同任何文档中的可打印章节。比如，考虑在一个bib文件中有如下条目：

```
@Book{companion,
  author    = {Michel Goossens and Frank Mittelbach and Alexander Samarin},
```

²⁵这里的 references 译为引文标注，指在引用命令导致在正文中出现的标注，这个标注由标签 label 构成。

根据plain.bst 样式, BibTeX 在bbl文件中输出该条目如下:

默认情况下，LaTeX 生成顺序制引用标签，因此**\bibitem**命令在aux文件中写入的行，如下：

要实现一个不同的引用标签样式，意味着需要通过aux文件传递更多的数据。比如，当使用natbib包时，aux文件包含的引用信息行，如下：

Biblatex 包支持任何格式的引用标签，因此引用命令需要访问所有的参考文献数据。只要看一看同样需要在引用中提供所有参考文献数据的jurabib包的输出，我们就可以明白到这对上述处理过程来说意味着什么。

这种情况下，整个thebibliography环境的内容通过aux文件有效传递。数据从bbl文件中读取出来，写入到aux中，再从aux读出保存到内存中。而参考文献表本身的生成仍然需要读入bbl文件。如果采用这种方式，Biblatex包将被迫通过aux文件回收所有的数据。但上述这个处理过程过度且多余，因为不管怎么样，数据都需要保存在内存中。

147

```

\documentclass{...}
\usepackage[...]{biblatex}
\addbibresource{...}
\begin{document}
\cite{...}
...
\printbibliography
\end{document}

```

为了简化整个流程，**Biblatex** 基本上以应用aux文件的方式应用bbl文件，并舍弃了**\bibcite**命令。于是，我们得到如下流程：

1. 运行**latex**: 第一步类似于上述的传统方式：**\bibstyle** 和 **\bibdata**以及所有引用的**\citation**命令写入到aux文件中 (以 **BibTeX** 为后端) 或者写到bcb文件中 (以 **Biber** 为后端)。然后等待后端提供需要的数据。当以 **BibTeX** 为后端时，**Biblatex** 使用一个特殊bst的文件，该文件用于实现 **BibTeX** 后端的数据接口，因此**\bibstyle**命令则必须是**\bibstyle{biblatex}**。
2. 运行**biber** 或 **bibtex**: 后端提供了辅助文件中所有**\citation**命令所需的条目，这些条目的数据来自bib文件。然而，它并不在bbl文件中写出一个可打印的参考文献表，而是一个参考文献的结构化表达数据。类似于aux文件，读入该bbl文件时不打印任何东西，仅是将数据存入内存中。
3. 运行**latex**: 第二次运行，bbl文件在文档正文开始的时候处理，类似于aux文件。从这开始，所有参考文献数据都已在内存中，所以所有的引用都可以正确打印。²⁶ 引用命令不仅可以访问预定义的标签，还可以访问完整的参考文献数据。参考文献表由内存中的相同数据生成，可以根据需要进行筛选和划分。

我们再次考虑上面给出的条目样例：

```

@Book{companion,
  author    = {Michel Goossens and Frank Mittelbach and Alexander Samarin},
  title     = {The LaTeX Companion},
  publisher = {Addison-Wesley},
  address   = {Reading, Mass.},
  year      = {1994},
}

```

使用 **Biblatex** 及 **Biber** 后端，这一条目实际上以如下格式输出：

²⁶如果**deferrednumbers**包选项打开，**Biblatex** 以类似于传统过程的一种算法来生成顺序制标签。这种情况下，这些数字在参考文献表打印的时候指定且需从后端辅助文件中回收。因此需要额外运行一次 **LaTeX** 以在标注中获得它们。


```

\entry{companion}{book}{}
  \labelname{author}{3}{}{%
    {{uniquename=0,hash=...}{Goossens}{G.}{Michel}{M.}{}{}{}{}{}%
    {{uniquename=0,hash=...}{Mittelbach}{M.}{Frank}{F.}{}{}{}{}{}%
    {{uniquename=0,hash=...}{Samarin}{S.}{Alexander}{A.}{}{}{}{}{}%
  }
  \name{author}{3}{}{%
    {{uniquename=0,hash=...}{Goossens}{G.}{Michel}{M.}{}{}{}{}{}%
    {{uniquename=0,hash=...}{Mittelbach}{M.}{Frank}{F.}{}{}{}{}{}%
    {{uniquename=0,hash=...}{Samarin}{S.}{Alexander}{A.}{}{}{}{}{}%
  }
  \list{publisher}{1}{}%
    {Addison-Wesley}%
  }
  \list{location}{1}{}%
    {Reading, Mass.}%
  }
  \field{title}{The LaTeX Companion}
  \field{year}{1994}
\endentry

```

由这一例子可见，某种程度上说结构化的数据构成了bbl文件内容²⁷。从这点上说，没有任何关于参考文献条目最终格式的决定。而参考文献表和引用标注的格式化由 LaTeX 宏控制，这些宏定义在参考文献和引用样式文件中。

4.2 参考文献样式 Bibliography Styles

一个参考文献样式是用于控制打印参考文献表中条目的宏的集合，定义在扩展名为bbx的文件中。Biblatex 包在其结尾加载所选择的参考文献样式文件。需要注意：一些由多个标准样式文件共享的常用宏定义在biblatex.def文件中。该文件同样在包结尾加载，但先于参考文献样式文件。

4.2.1 参考文献样式文件 Bibliography Style Files

在我们讨论参考文献样式的各部分之前，考虑一个典型的bbx文件总体结构，如下：

```

\ProvidesFile{example.bbx}[2006/03/15 v1.0 biblatex bibliography style]

\defbibenvironment{bibliography}
  {...}
  {...}
  {...}

```

²⁷这里应该是 bbl 文件而不是原文的 bib 文件

```

\defbibenvironment{shorthand}
{...}
{...}
{...}
\InitializeBibliographyStyle{...}
\DeclareBibliographyDriver{article}{...}
\DeclareBibliographyDriver{book}{...}
\DeclareBibliographyDriver{inbook}{...}
...
\DeclareBibliographyDriver{shorthand}{...}
\endinput

```

参考文献样式文件的主要结构包含如下命令:

`\RequireBibliographyStyle{<style>}`

该命令是可选的, 用于指定一些建立在更一般的参考文献样式上的特殊样式。该命令加载样式文件 `style.bbx`。

`\InitializeBibliographyStyle{<code>}`

该命令在参考文献表开始之前插入任意给定的 `<code>`, 但在参考文献表所形成的组内。该命令是可选的。它对于不同的参考文献驱动共享一些定义是有用的, 但不能用于参考文献组外。记住, 文档中可以有多个参考文献表, 如果参考文献驱动进行了任何全局设置, 应在下一个参考文献开始前重设²⁸。

`\DeclareBibliographyDriver{<entrytype>}{<code>}`

定义一个参考文献驱动。一个驱动 ‘driver’ 是一个宏用于控制某一具体的参考文献条目 (当打印参考文献表的时候) 或者某一具体命名了的参考文献表 (当打印多个参考文献表的时候)。`<entrytype>` 与 bib 文件中使用的条目类型对应, 以小写字母给出 (见 § 2.1)。`<entrytype>` 变量可以是一个星号。这种情况下, 该驱动退化为没有具体驱动的条目类型。`<code>` 是任意代码用于打印各自 `<entrytype>` 的参考文献条目。该命令是必须的。每个参考文献样式都应提供所用到的每类条目的驱动。

`\DeclareBibliographyAlias{<alias>}{<entrytype>}`

如果一个参考文献驱动用于处理多个参考文献条目类型, 该命令可以用来定义某类已经定义驱动的 `<entrytype>` 别名。`<alias>` 选项可以是一个星号, 这种情况下, 该驱动用于那些没有指定驱动的参考文献条目。

`\DeclareBibliographyOption{<key>}[<value>]{<code>}`

该命令以 `<key>=<value>` 格式定义额外的导言区选项。`<key>` 是选项键名。`<code>` 是任意 TeX 代码用于该选项使用时执行。传递给选项的 `value` 值作为 `<code>` 的输入参数 #1。可选的 `<value>` 是没有给定任何输入参数时使用的默认参数。这对于布尔开关是有用的, 比如, 某一选项定义如下:

²⁸这里不是很理解

```
\DeclareBibliographyOption{somekey}[true]{...}
```

当无键值给出时，该‘somekey’等价于‘somekey=true’。

```
\DeclareEntryOption{<key>}[<value>]{<code>}
```

类似于\DeclareBibliographyOption，但用于定义一些条目的可设选项，这些选项在 § 2.2.3 节中的options域中。<code>在 Biblatex 为一个引用命令或参考文献驱动准备条目数据时执行。

4.2.2 参考文献环境 Bibliography Environments

除了定义参考文献驱动，参考文献样式也要定义参考文献环境用于控制参考文献表的输出。这些环境由命令\defbibenvironment名义。默认情况下，\printbibliography使用bibliography环境。下面是一个适用于不打印标签的参考文献样式的环境定义：

```
\defbibenvironment{bibliography}
{
  \list
  {}
  {\setlength{\leftmargin}{\bibhang}%
   \setlength{\itemindent}{-\leftmargin}%
   \setlength{\itemsep}{\bibitemsep}%
   \setlength{\parsep}{\bibparsep}}
  {\endlist}
  {\item}
```

该定义使用 Biblatex 提供的\bibhang尺寸，应用了一个带悬挂缩进的list环境。它允许使用\bibitemsep 和 \bibparsep来实现一定程度的布局调整，Biblatex 提供的这两个尺寸就是为了该目的（见 § 4.10.3）。作者年制 (authoryear) 和作者题名制 (authortitle) 的参考文献样式使用类似于该例的定义。

```
\defbibenvironment{bibliography}
{
  \list
  {\printfield[labelnumberwidth]{labelnumber}}
  {\setlength{\labelwidth}{\labelnumberwidth}%
   \setlength{\leftmargin}{\labelwidth}%
   \setlength{\labelsep}{\biblabelsep}%
   \addtolength{\leftmargin}{\labelsep}%
   \setlength{\itemsep}{\bibitemsep}%
   \setlength{\parsep}{\bibparsep}}%
  \renewcommand*{\makelabel}[1]{\hss##1}
  {\endlist}
  {\item}
```

一些参考文献样式在参考文献列表中打印标签。比如,设计一个顺序引用格式的参考文献样式需要在参考文献表的每个条目前面打印顺序数字,这样参考文献看起来就像一个顺序列表。在第一个例子中,`\list`命令的第一个参数是空的。在这个例子中,我们需要在其中插入数字,这些数字由 Biblatex 的 `labelnumber` 域中的数字提供。我们也应用 Biblatex 提供的几个尺寸和工具,详见 §§ 4.10.4 and 4.10.5。顺序制 (numeric) 参考文献样式使用如上的定义。除 `labelnumber` 由 `labelalpha` 代替和 `labelnumberwidth` 由 `labelalphawidth` 代替外,顺序字母制 (alphabetic) 的样式也是类似的。

各参考文献表以相似方式处理。`\printbiblist`命令默认使用以 `bibliography list` 命名的环境 (当使用 BibTeX 时,`\printshorthands`总是使用 `shorthand` 环境)。一个典型的例子如下,其中的尺寸和工具定义详见第 §§ 4.10.4 and 4.10.5 节。

```
\defbibenvironment{shorthand}
{
  \list
    {
      \printfield{shorthandwidth}{shorthand}}
    {\setlength{\labelwidth}{\shorthandwidth}%
     \setlength{\leftmargin}{\labelwidth}%
     \setlength{\labelsep}{\biblabelsep}%
     \addtolength{\leftmargin}{\labelsep}%
     \setlength{\itemsep}{\bibitemsep}%
     \setlength{\parsep}{\bibparsep}%
     \renewcommand*{\makelabel}[1]{##1\hss}}
  {\endlist}
  {\item}
```

4.2.3 参考文献驱动 Bibliography Drivers

在我们讨论 Biblatex 包的数据接口命令前,了解一下参考文献驱动的结构是有益的。注意,虽然下面给出的例子是大大简化的,但仍具有说明意义。为可读性考虑,我们忽略了一些可能是 `@book` 条目的域,并且简化处理没有忽略的域。主要是为了说明驱动的结构。关于 BibTeX 文件的格式域与 Biblatex 包的数据类型的映射信息,见 § 2.2。

```
\DeclareBibliographyDriver{book}{%
  \printnames{author}%
  \newunit\newblock
  \printfield{title}%
  \newunit\newblock
  \printlist{publisher}%
  \newunit
  \printlist{location}%
  \newunit}
```

```
\printfield{year}%
\finentry}
```

标准的参考文献样式应用两个参考文献宏 `begentry` 和 `finentry`。

```
\DeclareBibliographyDriver{entrytype}{%
  \usebibmacro{begentry}
  ...
  \usebibmacro{finentry}}
```

作为默认的定义。

```
\newbibmacro*{begentry}{}
\newbibmacro*{finentry}{\finentry}
```

推荐使用这两个宏，因为方便在驱动开始或结束时使用钩子。

回到上述给出 `book` 条目类型的驱动，我们发现有一些缺省：即 `\printnames`，`\printlist`，和 `\printfield` 命令所使用的格式命令。为了说明一个格式话指令是什么，这里给出上述驱动举例中所使用虚构指令。域的格式是直接的，域的值直接作为参数传递给格式命令，并根据需要格式化。下面的指令简单地将输入参数用一个 `\emph` 命令包裹：

```
\DeclareFieldFormat{title}{\emph{#1}}
```

List formats are slightly more complex. After splitting up the list into individual items, Biblatex will execute the formatting directive once for every item in the list. The item is passed to the directive as an argument. The separator to be inserted between the individual items in the list is also handled by the corresponding directive, hence we have to check whether we are in the middle of the list or at the end when inserting it.

列表格式则要复杂一些。在将列表划分为独立的项后，Biblatex 将对列表中的每一项执行格式化命令。各项作为参数传递给格式化命令。列表中各项间的分隔符由相应的命令控制，因此我们必须在插入分隔符前要检查是否在列表中或者是列表末尾。

```
\DeclareListFormat{location}{%当打印的项的序号小于最大序号说明在列表内
#1%所以用逗号隔开
\ifthenelse{\value{listcount}<\value{liststop}}
{\addcomma\space}
{}}
```

姓名 (name) 的格式化命令类似于这种抄录列表，但列表中的单个项是姓名，因此需要自动的解析为姓名的不同组成部分。列表格式化命令对列表中每个姓名都执行一次，信吗的各个部分以分开的参数传递给该命令。比如，`#1` 是姓 (last name) 和 `#3` 是名 (first name)。下面给出一个简化的格式化命令例子：

Formatting directives for names are similar to those for literal lists, but the individual items in the list are names which are automatically dissected into their components.

The list formatting directive is executed once for each name in the list and the components of the name are passed to the formatting directive as separate arguments. For example, #1 is the last name and #3 is the first name. Here is a simplified example:

```
\DeclareNameFormat{author}{%
  \ifthenelse{\value{listcount}=1}
  {#1%
    \ifblank{#3}{}{\addcomma\space #3}}
  {\ifblank{#3}{}{#3\space}%
    #1}%
  \ifthenelse{\value{listcount}<\value{liststop}}
  {\addcomma\space}
  {}}
```

上述各格式化命令调换了第一个作者的姓名前后顺序“Last, First”，而其余姓名则是常规顺序（“First Last”）。注意：必须要保证提供的姓名部分是姓（last name），因此我们必须检查实际数据中姓名的哪些部分是存在的。如果姓名的一些部分不存在，则相关的变量就为空。如同抄录列表的命令，在各独立项之间插入的分隔符也由格式化命令控制，因为我们也要检查是否处于列表中还是在其末尾，这也是第二个\ifthenelse命令做的事情。

The above directive reverses the name of the first author (“Last, First”) and prints the remaining names in their regular sequence (“First Last”). Note that the only component which is guaranteed to be available is the last name, hence we have to check which parts of the name are actually present. If a certain component of a name is not available, the corresponding argument will be blank. As with directives for literal lists, the separator to be inserted between the individual items in the list is also handled by the formatting directive, hence we have to check whether we are in the middle of the list or at the end when inserting it. This is what the second \ifthenelse test does.

4.2.4 特殊域 Special Fields

下面的列表和域用于 Biblatex 传递数据给参考文献驱动和引用命令。它们由包自动定义，并不是用于bib文件。从参考文献和引用样式的角度，它们与bib中的域并没有什么不同。

The following lists and fields are used by Biblatex to pass data to bibliography drivers and citation commands. They are not used in bib files but defined automatically by the package. From the perspective of a bibliography or citation style, they are not different from the fields in a bib file.

4.2.4.1 一般域 Generic Fields

entrykey field (string)

bib文件中某一项的 entry key。这是一个字符串，用于 Biblatex 及其后端确定bib文件中的一个条目。

The entry key of an item in the `bib` file. This is the string used by Biblatex and the backend to identify an entry in the `bib` file.

`childentrykey` field (string)

当引用一个条目集的子条目时，Biblatex 给引用数据提供了父@set条目的数据。这意味着`entrykey`表示的是父条目的关键词。而子条目的关键词在`childentrykey`域中提供。该域仅在引用一个条目集的某一子条目时使用。

When citing a subentry of an entry set, Biblatex provides the data of the parent @set entry to citation commands. This implies that the `entrykey` field holds the entry key of the parent. The entry key of the child entry being cited is provided in the `childentrykey` field. This field is only available when citing a subentry of an entry set.

`datelabelsource` field (literal)

Biber only

该域决定由`\DeclareLabeldate`命令所选择的 `date` 域名的‘date’的前缀词。比如，如果标签的 `date` 域是`eventdate`，那么`datelabelsource`将会是‘event’。如果`\DeclareLabeldate`命令选择`date`域，那么`datelabelsource`将会定义一个空字符串。这使得`datelabelsource`的内容可以用于构建对`\DeclareLabeldate`选择的域的引用。因为`\DeclareLabeldate`也能选择用于备用的抄录字符串，如果`datelabelsource`未定义，那么要不`labeldate`包选项设置为 `false`，或是`\DeclareLabeldate`命令选择一个抄录字符串用于代替 `date` 域。要记住`\DeclareLabeldate`也用来选择非日期域作为备用，因此`datelabelsource`可以包含一个域名。所以，总结起来，规则是：

Holds the prefix coming before ‘date’ of the date field name chosen by `\DeclareLabeldate`. For example, if the label date field is `eventdate`, then `datelabelsource` will be ‘event’. In case `\DeclareLabeldate` selects the date field, then `datelabelsource` will be defined but will be an empty string as the prefix coming before ‘date’ in the date label name is empty. This is so that the contents of `datelabelsource` can be used in constructing references to the field which `\DeclareLabeldate` chooses. Since `\DeclareLabeldate` can also select literal strings for fallbacks, if `datelabelsource` is undefined, then either the `labeldate` package option is set to `false` or `\DeclareLabeldate` chose a literal string instead of a date field. Bear in mind that `\DeclareLabeldate` can also be used to select non-date fields as a fallback and so `datelabelsource` might contain a field name. So, in summary, the rules are

```
\iffieldundef{datelabelsource}
{
% labeldate package option is not set or \DeclareLabeldate resolved t
}
{
```

```

\iffielddundef{\thefield{datelabelsource}}
{
% datelabelsource contains a date field name prefix like "", "event",
}
{
% datelabelsource contains a non-date field
}
}

```

entrytype field (string)

条目类型 (@book, @inbook, 等), 以小写字母给出。

The entry type (@book, @inbook, etc.), given in lowercase letters.

childentrytype field (string)

当引用一个条目集的子条目时, Biblatex 给引用命令提供父集条目的数据。这意味着 entrytype 保存父条目的类型。子条目的类型则由 childentrytype 域提供。该域仅在引用一个条目集的子条目时使用。

When citing a subentry of an entry set, Biblatex provides the data of the parent @set entry to citation commands. This implies that the entrytype field holds the entry type of the parent. The entry type of the child entry being cited is provided in the childentrytype field. This field is only available when citing a subentry of an entry set.

entrysetcount field (integer)

该域保存的整数用于指明一个集中某个集成员的位置 (起始值是 1)。该域仅对一个条目集的子条目有用。

This field holds an integer indicating the position of a set member in the entry set (starting at 1). This field is only available in the subentries of an entry set.

hash field (string)

Biber only

该域非常特殊, 仅在姓名格式化命令中使用。它保存一个 hash 字符串, 用于唯一的确定姓名列表中的单个姓名。该信息对于姓名列表中的所有姓名都有提供。另外参见 namehash 和 fullhash。

This field is special in that it is only available locally in name formatting directives. It holds a hash string which uniquely identifies individual names in a name list. This information is available for all names in all name lists. See also namehash and fullhash.

namehash field (string)

一个 hash 字符串用于唯一确定 labelname 列表。这对再现检查很有用。比如, 一个将再次出现的作者和编者用一个类似 'idem' 的字符串代替的引用样式, 可

以用`\savefield`命令保存`namehash`域,并将其用于后面`\iffieldequals`(见 §§ 4.6.1 and 4.6.2) 命令的比较中。`namehash`域通过`labelname`列表的截短得到,即它的结果与`maxnames`和`minnames`选项相关。另外参见`hash`和`fullhash`。

A hash string which uniquely identifies the `labelname` list. This is useful for recurrence checks. For example, a citation style which replaces recurrent authors or editors with a string like ‘idem’ could save the `namehash` field with `\savefield` and use it in a comparison with `\iffieldequals` later (see §§ 4.6.1 and 4.6.2). The `namehash` is derived from the truncated `labelname` list, i. e., it is responsive to `maxnames` and `minnames`. See also `hash` and `fullhash`.

fullhash field (string)

一个 `hash` 字符串用于唯一确定`labelname`列表。该域`namehash`有两点不同:1. 产生 `hash` 时忽略`shortauthor`和`shorteditor`列表。2. 该 `hash` 指的是完整的列表, 忽略`maxnames`和`minnames`选项。另外见`hash`和`namehash`。

A hash string which uniquely identifies the `labelname` list. This field differs from `namehash` in two details: 1) The `shortauthor` and `shorteditor` lists are ignored when generating the hash. 2) The hash always refers to the full list, ignoring `maxnames` and `minnames`. See also `hash` and `namehash`.

pageref list (literal)

如果`backref`包选项打开, 该域保存各被引用条目所在页的页码。如果文档中有`refsection`环境, 反向引用是针对当前参考文献节的。

If the `backref` package option is enabled, this list holds the page numbers of the pages on which the respective bibliography entry is cited. If there are `refsection` environments in the document, the back references are local to the reference sections.

sortinit field (literal)

该域保存用于排序的信息首字符。使用 BibTeX 时, 该域也用来代替`sortinithash`域。

This field holds the initial character of the information used during sorting. With BibTeX, this field is also used internally instead of `sortinithash`.

sortinithash field (string)

Biber only

使用 Biber 时, 该域保存排序字符串的第一个扩展字素集群 (基本上是第一个字符) 的 Unicode 排序规则算法主要权重的 `hash` 值。当按照字母表顺序划分参考文献列表时很有用, 该域有内部`\bibinitsep`所使用。(见 § 3.9.3)

With Biber, this field holds a hash of the (locale-specific) Unicode Collation Algorithm primary weight of the first extended grapheme cluster (essentially the first character) of the string used during sorting. This is useful when subdividing the bibliography alphabetically and is used internally by `\bibinitsep` (see § 3.9.3).

该域保存复制条目源条目的关键词。复制条目常用于处理相关条目和related域。This field holds the entry key of the entry from which an entry was cloned. Clones are created for entries which are mentioned in related fields as part of related entry processing, for example.

4.2.4.2 引用标签中使用的域 Fields for Use in Citation Labels

labelalpha field (literal)

当使用 BibTeX 为后端时，生成一个类似于传统 BibTeX 的alpha.bst 样式的标签。这一默认标签由抽取labelname列表的首字母加上出版年的最后两个数字构成。label域可用来重写它的非数值部分 (non-numeric portion)。如果定义了label域，Biblatex 将使用它的值加上出版年的后两个数字生成labelalpha。shorthand域也可用来重写整个标签。如果定义了该域，labelalpha就是shorthand域，而不是一个自动生成的标签。使用 Biber 时，用户可以定义用来构建字母顺序标签的模板 (见 § 4.5.4)，而默认的模板域上面 bibtex 后端使用的格式相同。一个完整的字母顺序 (‘alphabetic’) 标签由以下域构成: labelalpha 加 extraalpha。注意labelalpha和 extraalpha域需要打开包选项 labelalpha (§ 3.1.2.3)。另外参见 extraalpha和 § 3.9.1的\labelalphaothers。

Biber only

When using BibTeX as the backend, a label similar to the labels generated by the alpha.bst style of traditional BibTeX. This default label consists of initials drawn from the labelname list plus the last two digits of the publication year. The label field may be used to override its non-numeric portion. If the label field is defined, Biblatex will use its value and append the last two digits of the publication year when generating labelalpha. The shorthand field may be used to override the entire label. If defined, labelalpha is the shorthand rather than an automatically generated label. With Biber, users can specify a template used to construct the alphabetic label (see § 4.5.4) and the default template mirrors the format mentioned for bibtex above. A complete ‘alphabetic’ label consists of the fields labelalpha plus extraalpha. Note that the labelalpha and extraalpha fields need to be requested with the package option labelalpha (§ 3.1.2.3). See also extraalpha as well as \labelalphaothers in § 3.9.1.

Biber only

extraalpha field (integer)

当参考文献中包含同一作者同一年出版的多个引文时，‘alphabetic’ 引用格式常需要一个额外的字母加入标签来区分。这种情况下extraalpha域保存一个整数可用命令 \mknumalph 转换成字母或以其他方式格式化。该域类似于在作者年 (author-year) 格式中extrayear的作用。完整的 ‘alphabetic’ 的标签由labelalpha 加 extraalpha构成。注意labelalpha和 extraalpha域需要打开包选项 labelalpha (详见 § 3.1.2.3)。另外参见 labelalpha和

§ 3.9.1的`\labelalphaothers`。表5总结了不同的`extra*`非歧义计数器和他们追踪的信息。

The ‘alphabetic’ citation scheme usually requires a letter to be appended to the label if the bibliography contains two or more works by the same author which were all published in the same year. In this case, the `extraalpha` field holds an integer which may be converted to a letter with `\mknumalph` or formatted in some other way. This field is similar to the role of `extrayear` in the author-year scheme. A complete ‘alphabetic’ label consists of the fields `labelalpha` plus `extraalpha`. Note that the `labelalpha` and `extraalpha` fields need to be requested with the package option `labelalpha`, see § 3.1.2.3 for details. See also `labelalpha` as well as `\labelalphaothers` in § 3.9.1. Table 5 summarises the various `extra*` disambiguation counters and what they track.

`labelname` list (name)

引用中打印的姓名。该列表可以是`shortauthor`, `author`, `shorteditor`, `editor`, 或`translator`域的复制值, 正常情况以该顺序检测。如果没有作者 (`authors`) 和编者 (`authors`), 该列表时未定义的。注意该列表也与`use<name>`相关, 见 § 3.1.3。引用样式打印引用中的姓名时使用这一列表。提供该列表仅为方便起见, 没有附加的意义。使用 Biber 时, 该域可以定制, 详见 § 4.5.9。

Biber only

The name to be printed in citations. This list is a copy of either the `shortauthor`, the `author`, the `shorteditor`, the `editor`, or the `translator` list, which are normally checked for in this order. If no authors and editors are available, this list is undefined. Note that this list is also responsive to the `use<name>`, options, see § 3.1.3. Citation styles should use this list when printing the name in a citation. This list is provided for convenience only and does not carry any additional meaning. With Biber, this field may be customized. See § 4.5.9 for details.

Biber only

`labelnumber` field (literal)

The number of the bibliography entry, as required by numeric citation schemes. If the `shorthand` field is defined, Biblatex does not assign a number to the respective entry. In this case `labelnumber` is the shorthand rather than a number. Numeric styles must use the value of this field instead of a counter. Note that this field needs to be requested with the package option `labelnumber`, see § 3.1.2.3 for details. Also see the package option `defernumbers` in § 3.1.2.1.

`prefixnumber` field (literal)

If the `prefixnumbers` option of `\printbibliography` has been set in order to prefix all entries in a subbibliography with a fixed string, this string is available in the `prefixnumber` field of all affected entries. If no prefix has been set, the `prefixnumber` field of the respective entry is undefined. See the `prefixnumbers` option of `\printbibliography` in § 3.6.2 for details. If the `shorthand` field is de-

finer, BibLaTeX does not assign the prefix to the `prefixnumber` field of the respective entry. In this case, the `prefixnumber` field is undefined.

`labeltitle` field (literal)

The printable title of a work. In some circumstances, a style might need to choose a title from a list of possible title fields. For example, citation styles printing short titles may want to print the `shorttitle` field if it exists but otherwise print the `title` field. The list of fields to be considered when constructing `labeltitle` may be customized. See § 4.5.9 for details. Note that the `extratitle` field needs to be requested with the package option `labeltitle`, see § 3.1.2.3 for details. See also `extratitle`. Note also that the `extratitleyear` field needs to be requested with the package option `labeltitleyear`. See also `extratitleyear`.

Biber only

`extratitle` field (integer)

It is sometimes useful, for example in author-title citation schemes, to be able to disambiguate works with the same title. For works by the same `labelname` with the same `labeltitle`, the `extratitle` field holds an integer which may be converted to a letter with `\mknumalph` or formatted in some other way (or it can be merely used as a flag to say that some other field such as a date should be used in conjunction with the `labeltitle` field). This field is undefined if there is only one work with the same `labeltitle` by the same `labelname` in the bibliography. Note that the `extratitle` field needs to be requested with the package option `labeltitle`, see § 3.1.2.3 for details. See also `labeltitle`. Table 5 summarises the various `extra*` disambiguation counters and what they track.

`extratitleyear` field (integer)

It is sometimes useful, for example in author-title citation schemes, to be able to disambiguate works with the same title in the same year but with no author. For works with the same `labeltitle` and with the same `labelyear`, the `extratitleyear` field holds an integer which may be converted to a letter with `\mknumalph` or formatted in some other way (or it can be merely used as a flag to say that some other field such as a publisher should be used in conjunction with the `labelyear` field). This field is undefined if there is only one work with the same `labeltitle` and `labelyear` in the bibliography. Note that the `extratitleyear` field needs to be requested with the package option `labeltitleyear`, see § 3.1.2.3 for details. See also `labeltitleyear`. Table 5 summarises the various `extra*` disambiguation counters and what they track.

`labelyear` field (literal)

The publication year, as specified in the `date` or the `year` field, for use in author-year labels. A complete author-year label consists of the fields `labelyear` plus `extrayear`. Note that the `labelyear` and `extrayear` fields need to be requested with the package option `labeldate`, see § 3.1.2.3 for details. See also `extrayear`. With Biber, the source date field for this may be customized. See § 4.5.9 for details.

Biber only

`labelmonth` field (datepart)

The publication month, as specified in the `date` or the `month` field, for use in author-year labels. Note that the `labelmonth` field needs to be requested with the package option `labeldate`, see § 3.1.2.3 for details. With Biber, the source date field for this may be customized. See § 4.5.9 for details.

Biber only

`labelday` field (datepart)

The publication day, as specified in the `date`, for use in author-year labels. Note that the `labelday` field needs to be requested with the package option `labeldate`, see § 3.1.2.3 for details. With Biber, the source date field for this may be customized. See § 4.5.9 for details.

Biber only

`extrayear` field (integer)

The author-year citation scheme usually requires a letter to be appended to the year if the bibliography contains two or more works by the same author which were all published in the same year. In this case, the `extrayear` field holds an integer which may be converted to a letter with `\mknumalph` or formatted in some other way. This field is undefined if there is only one work by the author in the bibliography or if all works by the author have different publication years. A complete author-year label consists of the fields `labelyear` plus `extrayear`. Note that the `labelyear` and `extrayear` fields need to be requested with the package option `labeldate`, see § 3.1.2.3 for details. See also `labelyear`. Table 5 summarises the various `extra*` disambiguation counters and what they track.

4.2.4.3 date 的分解域 Date Component Fields See table 8 for an overview of how the date fields in `bib` files are related to the date fields provided by the style interface. When testing for a field like `origdate` in a style, use code like:

```
\iffielddundef{origyear}{...}{...}
```

This will tell you if the corresponding date is defined at all. This test:

```
\iffielddundef{origendyear}{...}{...}
```

will tell you if the corresponding date is defined and a (fully specified) range. This test:

```
\iffieldequalstr{origendyear}{}{...}{...}
```

will tell you if the corresponding date is defined and an open-ended range. Open-ended ranges are indicated by an empty `endyear` component (as opposed to an undefined `endyear` component). See § 2.3.8 and table 3 on page 42 for further examples.

`day` field (datepart)

This field holds the day component of the `date` field. If the date is a range, it holds the day component of the start date.

bib File		Data Interface	
Field	Value (Example)	Field	Value (Example)
date	1988	day	undefined
		month	undefined
		year	1988
		endday	undefined
		endmonth	undefined
date	1997/	endyear	undefined
		day	undefined
		month	undefined
		year	1997
		endday	undefined
urldate	2009-01-31	endmonth	undefined
		endyear	empty
		urlday	31
		urlmonth	01
		urlyear	2009
origdate	2002-01/2002-02	urlendday	undefined
		urlendmonth	undefined
		urlendyear	undefined
		origday	undefined
		origmonth	01
eventdate	1995-01-31/1995-02-05	origyear	2002
		origendday	undefined
		origendmonth	02
		origendyear	2002
		eventday	31
		eventmonth	01
		eventyear	1995
		eventendday	05
		eventendmonth	02
		eventendyear	1995

Table 8: Date Interface

`month` field (datepart)

This field is the `month` as given in the database file or it holds the month component of the `date` field. If the date is a range, it holds the month component of the start date.

`year` field (datepart)

This field is the `year` as given in the database file or it holds the year component of the `date` field. If the date is a range, it holds the year component of the start date.

`endday` field (datepart)

If the date specification in the `date` field is a range, this field holds the day component of the end date.

`endmonth` field (datepart)

If the date specification in the `date` field is a range, this field holds the month component of the end date.

`endyear` field (datepart)

If the date specification in the `date` field is a range, this field holds the year component of the end date. A blank (but defined) `endyear` component indicates an open ended date range.

`origday` field (datepart)

This field holds the day component of the `origdate` field. If the date is a range, it holds the day component of the start date.

`origmonth` field (datepart)

This field holds the month component of the `origdate` field. If the date is a range, it holds the month component of the start date.

`origyear` field (datepart)

This field holds the year component of the `origdate` field. If the date is a range, it holds the year component of the start date.

`origendday` field (datepart)

If the date specification in the `origdate` field is a range, this field holds the day component of the end date.

`origendmonth` field (datepart)

If the date specification in the `origdate` field is a range, this field holds the month component of the end date.

`origendyear` field (datepart)

If the date specification in the `origdate` field is a range, this field holds the year component of the end date. A blank (but defined) `origendyear` component indicates an open ended `origdate` range.

`eventday` field (datepart)

This field holds the day component of the `eventdate` field. If the date is a range, it holds the day component of the start date.

`eventmonth` field (datepart)

This field holds the month component of the `eventdate` field. If the date is a range, it holds the month component of the start date.

`eventyear` field (datepart)

This field holds the year component of the `eventdate` field. If the date is a range, it holds the year component of the start date.

`eventendday` field (datepart)

If the date specification in the `eventdate` field is a range, this field holds the day component of the end date.

`eventendmonth` field (datepart)

If the date specification in the `eventdate` field is a range, this field holds the month component of the end date.

`eventendyear` field (datepart)

If the date specification in the `eventdate` field is a range, this field holds the year component of the end date. A blank (but defined) `eventendyear` component indicates an open ended `eventdate` range.

`urlday` field (datepart)

This field holds the day component of the `urldate` field.

`urlmonth` field (datepart)

This field holds the month component of the `urldate` field.

`urlyear` field (datepart)

This field holds the year component of the `urldate` field.

`urlendday` field (datepart)

If the date specification in the `urldate` field is a range, this field holds the day component of the end date.

`urlendmonth` field (datepart)

If the date specification in the `urldate` field is a range, this field holds the month component of the end date.

`urlendyear` field (datepart)

If the date specification in the `urldate` field is a range, this field holds the year component of the end date. A blank (but defined) `urlendyear` component indicates an open ended `urldate` range.

4.3 标注样式 Citation Styles

标注样式是诸如`\cite`等用于打印不同类型标注的命令集。这些样式定义在后缀为`cbx`的文件中。Biblatex 在包末尾加载它们。注意: 一些标准标注样式所共享的常用的宏集在`biblatex.def`文件中。这一文件也再包末尾加载, 先于选择的标注样式。它也包含来自 § 3.7.5 节的命令的定义。

4.3.1 标注样式文件 Citation Style Files

在讨论标注样式文件提供的各个命令前, 考虑如下一个典型的`cbx`文件的整体结构:

```
\ProvidesFile{example.cbx}[2006/03/15 v1.0 biblatex citation style]

\DeclareCiteCommand{\cite}{...}{...}{...}{...}
\DeclareCiteCommand{\parencite}[\mkbibparens]{...}{...}{...}{...}
\DeclareCiteCommand{\footcite}[\mkbibfootnote]{...}{...}{...}{...}
\DeclareCiteCommand{\textcite}{...}{...}{...}{...}
\endinput
```

`\RequireCitationStyle{<style>}`

这个命令是可选的, 用于在一些更一般的样式基础上构建特殊的标注样式。它加载标注样式`style.cbx`。

`\InitializeCitationStyle{<code>}`

指定初始化或重设标注样式需要的任意`<code>`。这个钩子将在包加载的时候执行一次, 并且每次都使用 § 3.7.8 节的`\citereset`命令。`\citereset`命令也重设本宏包的内部标注追踪器。它会影响 § 4.6.2 节中列出的 `cmdifcitereen`, `\ifentryseen`, `\ifciteibid`, 和 `\ifciteidem` 等判断。当使用 `refsection` 环境时, 标注追踪器重设当前的 `refsection` 局部环境。

`\OnManualCitation{⟨code⟩}` 指定标注样式部分重设需要的任意⟨code⟩。这一钩子将在 § 3.7.8 中的 `\mancite` 命令使用时调用。它有时特别有用，可以代替像 ‘*ibidem*’ 或 ‘*op. cit.*’ 等缩写表示的重复标注，因为当自动生成和人工产生的标注混合使用的时候这些缩写可能会有歧义。`\mancite` 命令也会重设宏包的内部 ‘*ibidem*’ 和 ‘*idem*’ 追踪器，进而影响 § 4.6.2 讨论的 `\ifc cite ibid` 和 `\ifc cite idem` 判断。

```
\DeclareCiteCommand{⟨command⟩}[⟨wrapper⟩]{⟨precode⟩}{⟨loopcode⟩}{⟨sepcode⟩}{⟨postcode⟩}
\DeclareCiteCommand*{⟨command⟩}[⟨wrapper⟩]{⟨precode⟩}{⟨loopcode⟩}{⟨sepcode⟩}{⟨postcode⟩}
```

This is the core command used to define all citation commands. It takes one optional and five mandatory arguments. The `⟨command⟩` is the command to be defined, for example `\cite`. If the optional `⟨wrapper⟩` argument is given, the entire citation will be passed to the `⟨wrapper⟩` as an argument, i.e., the wrapper command must take one mandatory argument.²⁹ The `⟨precode⟩` is arbitrary code to be executed at the beginning of the citation. It will typically handle the `⟨prenote⟩` argument which is available in the `prenote` field. It may also be used to initialize macros required by the `⟨loopcode⟩`. The `⟨loopcode⟩` is arbitrary code to be executed for each entry key passed to the `⟨command⟩`. This is the core code which prints the citation labels or any other data. The `⟨sepcode⟩` is arbitrary code to be executed after each iteration of the `⟨loopcode⟩`. It will only be executed if a list of entry keys is passed to the `⟨command⟩`. The `⟨sepcode⟩` will usually insert some kind of separator, such as a comma or a semicolon. The `⟨postcode⟩` is arbitrary code to be executed at the end of the citation. The `⟨postcode⟩` will typically handle the `⟨postnote⟩` argument which is available in the `postnote` field.³⁰ The starred variant of `\DeclareCiteCommand` defines a starred `⟨command⟩`. For example, `\DeclareCiteCommand*{cite}` would define `\cite*`.³¹

```
\DeclareMultiCiteCommand{⟨command⟩}[⟨wrapper⟩]{⟨cite⟩}{⟨delimiter⟩}
```

This command defines ‘*multicite*’ commands (§ 3.7.3). The `⟨command⟩` is the *multicite* command to be defined, for example `\cites`. It is automatically made robust. *Multicite* commands are built on top of backend commands defined with `\DeclareCiteCommand` and the `⟨cite⟩` argument specifies the name of the backend command to be used. Note that the wrapper of the backend command (i.e., the `⟨wrapper⟩` argument passed to `\DeclareCiteCommand`) is ignored. Use the optional `⟨wrapper⟩` argument to specify an alternative wrapper. The `⟨delimiter⟩` is the string to be printed as a separator between the individual citations in the list. This will typically be `\multicitedelim`. The following examples are real definitions taken from `biblatex.def`:

²⁹Typical examples of wrapper commands are `\mkbibparens` and `\mkbibfootnote`.

³⁰The bibliographic data available to the `⟨loopcode⟩` is the data of the entry currently being processed. In addition to that, the data of the first entry is available to the `⟨precode⟩` and the data of the last one is available to the `⟨postcode⟩`. ‘*First*’ and ‘*last*’ refer to the order in which the citations are printed. If the `sortcites` package option is active, this is the order of the list after sorting. Note that no bibliographic data is available to the `⟨sepcode⟩`.

³¹Note that the regular variant of `\DeclareCiteCommand` defines a starred version of the `⟨command⟩` implicitly, unless the starred version has been defined before. This is intended as a fallback. The implicit definition is an alias for the regular variant.

```

\DeclareMultiCiteCommand{\cites}%
    {\cite}{\multicitedelim}
\DeclareMultiCiteCommand{\parencites}[\mkbibparens]%
    {\parencite}{\multicitedelim}
\DeclareMultiCiteCommand{\footcites}[\mkbibfootnote]%
    {\footcite}{\multicitedelim}

```

```

\DeclareAutoCiteCommand{<name>}[<position>]{<cite>}{<multicite>}

```

This command provides definitions for the `\autocite` and `\autocites` commands from § 3.7.4. The definitions are enabled with the `autocite` package option from § 3.1.2.1. The `<name>` is an identifier which serves as the value passed to the package option. The autocite commands are built on top of backend commands like `\parencite` and `\parencites`. The arguments `<cite>` and `<multicite>` specify the backend commands to use. The `<cite>` argument refers to `\autocite` and `<multicite>` refers to `\autocites`. The `<position>` argument controls the handling of any punctuation marks after the citation. Possible values are `l`, `r`, `f`. `r` means that the punctuation is placed to the right of the citation, i.e., it will not be moved around. `l` means that any punctuation after the citation is moved to the left of the citation. `f` is like `r` in a footnote and like `l` otherwise. This argument is optional and defaults to `r`. See also `\DeclareAutoPunctuation` in § 4.7.5 and the `autopunct` package option in § 3.1.2.1. The following examples are real definitions taken from `biblatex.def`:

```

\DeclareAutoCiteCommand{plain}{\cite}{\cites}
\DeclareAutoCiteCommand{inline}{\parencite}{\parencites}
\DeclareAutoCiteCommand{footnote}[l]{\footcite}{\footcites}
\DeclareAutoCiteCommand{footnote}[f]{\smartcite}{\smartcites}

```

A definition provided in the document preamble can be subsequently adopted with the following: (see § 3.2.2).

```

\ExecuteBibliographyOptions{autocite=name}

```

4.3.2 特殊域 Special Fields

The following fields are used by Biblatex to pass data to citation commands. They are not used in `bib` files but defined automatically by the package. From the perspective of a citation style, they are not different from the fields in a `bib` file. See also § 4.2.4.

`prenote` field (literal)

The `<prenote>` argument passed to a citation command. This field is specific to citations and not available in the bibliography. If the `<prenote>` argument is missing or empty, this field is undefined.

`postnote` field (literal)

The $\langle postnote \rangle$ argument passed to a citation command. This field is specific to citations and not available in the bibliography. If the $\langle postnote \rangle$ argument is missing or empty, this field is undefined.

`multiprenote` field (literal)

The $\langle multiprenote \rangle$ argument passed to a multicite command. This field is specific to citations and not available in the bibliography. If the $\langle multiprenote \rangle$ argument is missing or empty, this field is undefined.

`multi postnote` field (literal)

The $\langle multi postnote \rangle$ argument passed to a multicite command. This field is specific to citations and not available in the bibliography. If the $\langle multi postnote \rangle$ argument is missing or empty, this field is undefined.

`postpunct` field (punctuation command)

The trailing punctuation argument implicitly passed to a citation command. This field is specific to citations and not available in the bibliography. If the character following a given citation command is not specified in `\DeclareAutoPunctuation` (§ 4.7.5), this field is undefined.

4.4 数据接口 Data Interface

The data interface are the facilities used to format and print all bibliographic data. These facilities are available in both bibliography and citation styles.

4.4.1 数据命令 Data Commands

This section introduces the main data interface of the Biblatex package. These are the commands doing most of the work, i. e., they actually print the data provided in lists and fields.

`\printfield`[$\langle format \rangle$]{ $\langle field \rangle$ }

This command prints a $\langle field \rangle$ using the formatting directive $\langle format \rangle$, as defined with `\DeclareFieldFormat`. If a type-specific $\langle format \rangle$ has been declared, the type-specific formatting directive takes precedence over the generic one. If the $\langle field \rangle$ is undefined, nothing is printed. If the $\langle format \rangle$ is omitted, `\printfield` tries using the name of the field as a format name. For example, if the `title` field is to be printed and the $\langle format \rangle$ is not specified, it will try to use the field format `title`.³² In this case, any type-specific formatting directive will also take precedence over the generic one. If all of these formats are undefined, it falls back to `default` as a last resort. Note that `\printfield` provides the name of the field currently being processed in `\currentfield` for use in field formatting directives.

³²In other words, `\printfield{title}` is equivalent to `\printfield[title]{title}`.

`\printlist` [*<format>*] [*<start>*–*<stop>*] {*<literal list>*}

This command loops over all items in a *<literal list>*, starting at item number *<start>* and stopping at item number *<stop>*, including *<start>* and *<stop>* (all lists are numbered starting at 1). Each item is printed using the formatting directive *<format>*, as defined with `\DeclareListFormat`. If a type-specific *<format>* has been declared, the type-specific formatting directive takes precedence over the generic one. If the *<literal list>* is undefined, nothing is printed. If the *<format>* is omitted, `\printlist` tries using the name of the list as a format name. In this case, any type-specific formatting directive will also take precedence over the generic one. If all of these formats are undefined, it falls back to `default` as a last resort. The *<start>* argument defaults to 1; *<stop>* defaults to the total number of items in the list. If the total number is greater than *<maxitems>*, *<stop>* defaults to *<minitems>* (see § 3.1.2.1). See `\printnames` for further details. Note that `\printlist` provides the name of the literal list currently being processed in `\currentlist` for use in list formatting directives.

`\printnames` [*<format>*] [*<start>*–*<stop>*] {*<name list>*}

This command loops over all items in a *<name list>*, starting at item number *<start>* and stopping at item number *<stop>*, including *<start>* and *<stop>* (all lists are numbered starting at 1). Each item is printed using the formatting directive *<format>*, as defined with `\DeclareNameFormat`. If a type-specific *<format>* has been declared, the type-specific formatting directive takes precedence over the generic one. If the *<name list>* is undefined, nothing is printed. If the *<format>* is omitted, `\printnames` tries using the name of the list as a format name. In this case, any type-specific formatting directive will also take precedence over the generic one. If all of these formats are undefined, it falls back to `default` as a last resort. The *<start>* argument defaults to 1; *<stop>* defaults to the total number of items in the list. If the total number is greater than *<maxnames>*, *<stop>* defaults to *<minnames>* (see § 3.1.2.1). If you want to select a range but use the default list format, the first optional argument must still be given, but is left empty:

```
\printnames[] [1-3] { ... }
```

One of *<start>* and *<stop>* may be omitted, hence the following arguments are all valid:

```
\printnames[...] [-1] { ... }  
\printnames[...] [2-] { ... }  
\printnames[...] [1-3] { ... }
```

If you want to override *<maxnames>* and *<minnames>* and force printing of the entire list, you may refer to the `listtotal` counter in the second optional argument:

```
\printnames[...] [-\value{listtotal}] { ... }
```

Whenever `\printrnames` and `\printlist` process a list, information concerning the current state is accessible by way of four counters: the `listtotal` counter holds the total number of items in the current list, `listcount` holds the number of the item currently being processed, `liststart` is the `\start` argument passed to `\printrnames` or `\printlist`, `liststop` is the `\stop` argument. These counters are intended for use in list formatting directives. `listtotal` may also be used in the second optional argument to `\printrnames` and `\printlist`. Note that these counters are local to list formatting directives and do not hold meaningful values when used anywhere else. For every list, there is also a counter by the same name which holds the total number of items in the corresponding list. For example, the `author` counter holds the total number of items in the `author` list. These counters are similar to `listtotal` except that they may also be used independently of list formatting directives. There are also `maxnames` and `minnames` as well as `maxitems` and `minitems` counters which hold the values of the corresponding package options. See § 4.10.5 for a complete list of such internal counters. Note that `\printrnames` provides the name of the name list currently being processed in `\currentname` for use in name formatting directives.

`\printtext` [`\format`] {`\text`} 该命令用于打印`\text`，可以是可打印的文本或者任意产生可打印文本的代码。它清除插入`\text`之前的标点缓存并且通知 Biblatex 打印文本已经插入。这保证了所有之前和之后的`\newblock`和`\newunit`命令能产生期望的作用。`\printfield`、`\printrnames`、`\bibstring`及其相关命令都这般自动处理 (见 § 4.8)。如果一个参考文献样式需要如实插入文本 (包括来自 §§ 4.7.3 and 4.7.4 的命令)，需要使用该命令来确保 `block` 和 `unit` 标点在 § 4.7.1 节中所述正常功能。可选参数`\format`指定一个域格式指令用于格式化`\text`。当需要把若干个域打印为某一格式的集合块，这就会很有用，比如把集合块用括号或引号包起来。如果声明了 `type-specific` 的格式指令，则覆盖设置的一般格式指令。如果`\format`缺省，那么`\text`如实输出 (原样打印)。更多实用细节见第 § 4.11.7 节。

`\printfile` [`\format`] {`\file`}

This command is similar to `\printtext` except that the second argument is a file name rather than literal text. The `\file` argument must be the name of a valid LaTeX file found in TeX's search path. `\printfile` will use `\input` to load this `\file`. If there is no such file, `\printfile` does nothing. The optional `\format` argument specifies a field formatting directive to be applied to the `\file`. If a type-specific `\format` has been declared, the type-specific formatting directive takes precedence over the generic one. If the `\format` is omitted, the `\file` is printed as is. Note that this feature needs to be enabled explicitly by setting the package option `loadfiles` from § 3.1.2.1. By default, `\printfile` will not input any files.

`\printdate` This command prints the date of the entry, as specified in the fields `date` or `month/year`. The date format is controlled by the package option `date` from § 3.1.2.1. Additional formatting (fonts etc.) may be applied by adjusting the field format `date`

(§ 4.10.4). Note that this command interfaces with the punctuation tracker. There is no need to wrap it in a `\printtext` command.

`\printdateextra` Similar to `\printdate` but incorporates the `extrayear` field in the date specification. This is useful for bibliography styles designed for author-year citations.

`\printdatelabel` Similar to `\printdate` but prints the date field determined by `\DeclareLabeldate`. The date format is controlled by the package option `datelabel` from § 3.1.2.1. Additional formatting may be applied by adjusting the field format `datelabel` (§ 4.10.4).

`\printdateextralabel` Similar to `\printdatelabel` but incorporates the `extrayear` field in the date specification. This is useful for bibliography styles designed for author-year citations.

`\printurldate` Similar to `\printdate` but prints the `urldate` of the entry. The date format is controlled by the package option `urldate` from § 3.1.2.1. Additional formatting may be applied by adjusting the field format `urldate` (§ 4.10.4).

`\printorigdate` Similar to `\printdate` but prints the `origdate` of the entry. The date format is controlled by the package option `origdate` from § 3.1.2.1. Additional formatting may be applied by adjusting the field format `origdate` (§ 4.10.4).

`\printeventdate` Similar to `\printdate` but prints the `eventdate` of the entry. The date format is controlled by the package option `eventdate` from § 3.1.2.1. Additional formatting may be applied by adjusting the field format `eventdate` (§ 4.10.4).

`\indexfield`[*⟨format⟩*]{*⟨field⟩*}

This command is similar to `\printfield` except that the *⟨field⟩* is not printed but added to the index using the formatting directive *⟨format⟩*, as defined with `\DeclareIndexFieldFormat`. If a type-specific *⟨format⟩* has been declared, it takes precedence over the generic one. If the *⟨field⟩* is undefined, this command does nothing. If the *⟨format⟩* is omitted, `\indexfield` tries using the name of the field as a format name. In this case, any type-specific formatting directive will also take precedence over the generic one. If all of these formats are undefined, it falls back to `default` as a last resort.

`\indexlist`[*⟨format⟩*][*⟨start⟩–⟨stop⟩*]{*⟨literal list⟩*}

This command is similar to `\printlist` except that the items in the list are not printed but added to the index using the formatting directive *⟨format⟩*, as defined with `\DeclareIndexListFormat`. If a type-specific *⟨format⟩* has been declared, the type-specific formatting directive takes precedence over the generic one. If the *⟨literal list⟩* is undefined, this command does nothing. If the *⟨format⟩* is omitted, `\indexlist` tries using the name of the list as a format name. In this case, any type-specific formatting directive will also take precedence over the generic one. If all of these formats are undefined, it falls back to `default` as a last resort.

`\indexnames[⟨format⟩][⟨start⟩–⟨stop⟩]{⟨name list⟩}`

This command is similar to `\printnames` except that the items in the list are not printed but added to the index using the formatting directive `⟨format⟩`, as defined with `\DeclareIndexNameFormat`. If a type-specific `⟨format⟩` has been declared, the type-specific formatting directive takes precedence over the generic one. If the `⟨name list⟩` is undefined, this command does nothing. If the `⟨format⟩` is omitted, `\indexnames` tries using the name of the list as a format name. In this case, any type-specific formatting directive will also take precedence over the generic one. If all of these formats are undefined, it falls back to `default` as a last resort.

`\entrydata{⟨key⟩}{⟨code⟩}`

`\entrydata*{⟨key⟩}{⟨code⟩}`

Data commands like `\printfield` normally use the data of the entry currently being processed. You may use `\entrydata` to switch contexts locally. The `⟨key⟩` is the entry key of the entry to use locally. The `⟨code⟩` is arbitrary code to be executed in this context. This code will be executed in a group. See § 4.11.6 for an example. Note that this command will automatically switch languages if the `autolang` package option is enabled. The starred version `\entrydata*` will clone all fields of the enclosing entry, using `field`, `counter`, and other resource names prefixed with the string ‘`saved`’. This is useful when comparing two data sets. For example, inside the `⟨code⟩` argument, the `author` field holds the author of entry `⟨key⟩` and the author of the enclosing entry is available as `savedauthor`. The `author` counter holds the number of names in the `author` field of `⟨key⟩`; the `savedauthor` counter refers to the author count of the enclosing entry.

`\entryset{⟨precode⟩}{⟨postcode⟩}`

This command is intended for use in bibliography drivers handling `@set` entries. It will loop over all members of the set, as indicated by the `entryset` field, and execute the appropriate driver for the respective set member. This is similar to executing the `\usedriver` command from § 4.6.4 for each set member. The `⟨precode⟩` is arbitrary code to be executed prior to processing each item in the set. The `⟨postcode⟩` is arbitrary code to be executed immediately after processing each item. Both arguments are mandatory in terms of the syntax but may be left empty. See § 4.11.1 for usage examples.

`\DeclareFieldInputHandler{⟨field⟩}{⟨code⟩}`

This command can be used to define a data input handler for `⟨field⟩` when it is read from the `.bibl`. Within the `⟨code⟩`, the macro `\NewValue` contains the value of the field. For example, to ignore the `volumes` field when it appears, you could do

```
\DeclareFieldInputHandler{volumes}{\def\NewValue{}}
```


Generally, you would want to use `\DeclareSourcemap` (see § 4.5.2) to remove and modify fields but this alternative method may be useful in some circumstances when the emphasis is on appearance rather than data since the `<code>` can be arbitrary TeX.

```
\DeclareListInputHandler{<list>}{<code>}
```

As `\DeclareFieldInputHandler` but for lists. Within the `<code>`, the macro `\NewValue` contains the value of the list and `\NewCount` contains the number of items in the list.

```
\DeclareNameInputHandler{<name>}{<code>}
```

As `\DeclareFieldInputHandler` but for names. Within the `<code>`, the macro `\NewValue` contains the value of the name, `\NewCount` contains the number of individual names in the name and `\NewOption` contains any per-name options passed in the `.bbl`.

4.4.2 格式化指令 Formatting Directives

This section introduces the commands used to define the formatting directives required by the data commands from § 4.4.1. Note that all standard formats are defined in `biblatex.def`.

```
\DeclareFieldFormat[<entrytype, ...>]{<format>}{<code>}
```

```
\DeclareFieldFormat*{<format>}{<code>}
```

Defines the field format `<format>`. This formatting directive is arbitrary `<code>` to be executed by `\printfield`. The value of the field will be passed to the `<code>` as its first and only argument. The name of the field currently being processed is available to the `<code>` as `\currentfield`. If an `<entrytype>` is specified, the format is specific to that type. The `<entrytype>` argument may be a comma-separated list of values. The starred variant of this command is similar to the regular version, except that all type-specific formats are cleared.

```
\DeclareListFormat[<entrytype, ...>]{<format>}{<code>}
```

```
\DeclareListFormat*{<format>}{<code>}
```

Defines the literal list format `<format>`. This formatting directive is arbitrary `<code>` to be executed for every item in a list processed by `\printlist`. The current item will be passed to the `<code>` as its first and only argument. The name of the literal list currently being processed is available to the `<code>` as `\currentlist`. If an `<entrytype>` is specified, the format is specific to that type. The `<entrytype>` argument may be a comma-separated list of values. Note that the formatting directive also handles the punctuation to be inserted between the individual items in the list. You need to check whether you are in the middle of or at the end of the list, i. e., whether `listcount` is smaller than or equal to `liststop`. The starred variant of this command is similar to the regular version, except that all type-specific formats are cleared.

```
\DeclareNameFormat[⟨entrytype, ...⟩]{⟨format⟩}{⟨code⟩}
\DeclareNameFormat*{⟨format⟩}{⟨code⟩}
```

Defines the name list format $\langle format \rangle$. This formatting directive is arbitrary $\langle code \rangle$ to be executed for every name in a list processed by `\printnames`. If an $\langle entrytype \rangle$ is specified, the format is specific to that type. The $\langle entrytype \rangle$ argument may be a comma-separated list of values. The individual parts of a name will be passed to the $\langle code \rangle$ as separate arguments. These arguments are as follows:

- #1 The last names. If a name consists of a single part only (for example, ‘Aristotle’), this part will be treated as the last name.
- #2 The last names, given as initials.
- #3 The first names. This argument also includes all middle names.
- #4 The first names, given as initials.
- #5 The name prefixes, for example von, van, of, da, de, del, della, etc. Note that name prefixes are referred to as the ‘von part’ of the name in the BibTeX file format documentation.
- #6 The name prefixes, given as initials.
- #7 The name affixes, for example ‘junior’, ‘senior’, ‘der Jüngere’, ‘der Ältere’, etc. Note that name affixes are referred to as the ‘junior part’ of the name in the BibTeX file format documentation.
- #8 The name affixes, given as initials.

If a certain part of a name is not available, the corresponding argument will be empty, hence you may use `\ifblank` tests to check for the individual parts of a name. The name of the name list currently being processed is available to the $\langle code \rangle$ as `\currentname`. Note that the formatting directive also handles the punctuation to be inserted between separate names and between the individual parts of a name. You need to check whether you are in the middle of or at the end of the list, i. e., whether `listcount` is smaller than or equal to `liststop`. See also § 3.12.4. The starred variant of this command is similar to the regular version, except that all type-specific formats are cleared.

```
\DeclareIndexFieldFormat[⟨entrytype, ...⟩]{⟨format⟩}{⟨code⟩}
\DeclareIndexFieldFormat*{⟨format⟩}{⟨code⟩}
```

Defines the field format $\langle format \rangle$. This formatting directive is arbitrary $\langle code \rangle$ to be executed by `\indexfield`. The value of the field will be passed to the $\langle code \rangle$ as its first and only argument. The name of the field currently being processed is available to the $\langle code \rangle$ as `\currentfield`. If an $\langle entrytype \rangle$ is specified, the format is specific to that type. The $\langle entrytype \rangle$ argument may be a comma-separated list of values. This command is similar to `\DeclareFieldFormat` except that the data handled by the $\langle code \rangle$ is not intended to be printed but written to the index. Note that `\indexfield`

will execute the $\langle code \rangle$ as is, i. e., the $\langle code \rangle$ must include `\index` or a similar command. The starred variant of this command is similar to the regular version, except that all type-specific formats are cleared.

```
\DeclareIndexListFormat[ $\langle entrytype, ... \rangle$ ]{ $\langle format \rangle$ }{ $\langle code \rangle$ }
\DeclareIndexListFormat*{ $\langle format \rangle$ }{ $\langle code \rangle$ }
```

Defines the literal list format $\langle format \rangle$. This formatting directive is arbitrary $\langle code \rangle$ to be executed for every item in a list processed by `\indexlist`. The current item will be passed to the $\langle code \rangle$ as its only argument. The name of the literal list currently being processed is available to the $\langle code \rangle$ as `\currentlist`. If an $\langle entrytype \rangle$ is specified, the format is specific to that type. The $\langle entrytype \rangle$ argument may be a comma-separated list of values. This command is similar to `\DeclareListFormat` except that the data handled by the $\langle code \rangle$ is not intended to be printed but written to the index. Note that `\indexlist` will execute the $\langle code \rangle$ as is, i. e., the $\langle code \rangle$ must include `\index` or a similar command. The starred variant of this command is similar to the regular version, except that all type-specific formats are cleared.

```
\DeclareIndexNameFormat[ $\langle entrytype, ... \rangle$ ]{ $\langle format \rangle$ }{ $\langle code \rangle$ }
\DeclareIndexNameFormat*{ $\langle format \rangle$ }{ $\langle code \rangle$ }
```

Defines the name list format $\langle format \rangle$. This formatting directive is arbitrary $\langle code \rangle$ to be executed for every name in a list processed by `\indexnames`. The name of the name list currently being processed is available to the $\langle code \rangle$ as `\currentname`. If an $\langle entrytype \rangle$ is specified, the format is specific to that type. The $\langle entrytype \rangle$ argument may be a comma-separated list of values. The parts of the name will be passed to the $\langle code \rangle$ as separate arguments. This command is very similar to `\DeclareNameFormat` except that the data handled by the $\langle code \rangle$ is not intended to be printed but written to the index. Note that `\indexnames` will execute the $\langle code \rangle$ as is, i. e., the $\langle code \rangle$ must include `\index` or a similar command. The starred variant of this command is similar to the regular version, except that all type-specific formats are cleared.

```
\DeclareFieldAlias[ $\langle entry type \rangle$ ]{ $\langle alias \rangle$ }[ $\langle format entry type \rangle$ ]{ $\langle format \rangle$ }
```

Declares $\langle alias \rangle$ to be an alias for the field format $\langle format \rangle$. If an $\langle entrytype \rangle$ is specified, the alias is specific to that type. The $\langle format entry type \rangle$ is the entry type of the backend format. This is only required when declaring an alias for a type-specific formatting directive.

```
\DeclareListAlias[ $\langle entry type \rangle$ ]{ $\langle alias \rangle$ }[ $\langle format entry type \rangle$ ]{ $\langle format \rangle$ }
```

Declares $\langle alias \rangle$ to be an alias for the literal list format $\langle format \rangle$. If an $\langle entrytype \rangle$ is specified, the alias is specific to that type. The $\langle format entry type \rangle$ is the entry type of the backend format. This is only required when declaring an alias for a type-specific formatting directive.

`\DeclareNameAlias` [*<entry type>*] {*<alias>*} [*<format entry type>*] {*<format>*}

Declares *<alias>* to be an alias for the name list format *<format>*. If an *<entrytype>* is specified, the alias is specific to that type. The *<format entry type>* is the entry type of the backend format. This is only required when declaring an alias for a type-specific formatting directive.

`\DeclareIndexFieldAlias` [*<entry type>*] {*<alias>*} [*<format entry type>*] {*<format>*}

Declares *<alias>* to be an alias for the field format *<format>*. If an *<entrytype>* is specified, the alias is specific to that type. The *<format entry type>* is the entry type of the backend format. This is only required when declaring an alias for a type-specific formatting directive.

`\DeclareIndexListAlias` [*<entry type>*] {*<alias>*} [*<format entry type>*] {*<format>*}

Declares *<alias>* to be an alias for the literal list format *<format>*. If an *<entrytype>* is specified, the alias is specific to that type. The *<format entry type>* is the entry type of the backend format. This is only required when declaring an alias for a type-specific formatting directive.

`\DeclareIndexNameAlias` [*<entry type>*] {*<alias>*} [*<format entry type>*] {*<format>*}

Declares *<alias>* to be an alias for the name list format *<format>*. If an *<entrytype>* is specified, the alias is specific to that type. The *<format entry type>* is the entry type of the backend format. This is only required when declaring an alias for a type-specific formatting directive.

4.5 自定义 Customization

4.5.1 关联条目 Related Entries

关联条目功能由如下部分构成:³³

- 可将条目中的特殊域用于建立和描述关系
- 可把本地化字符串作为关联数据的前缀 (可选)
- 提供用于抽取和打印关联数据的宏
- 提供给本地化字符串和关联数据格式化的格式

特殊域是 `related`, `relatedtype`, `relatedstring` 和 `relatedoptions`:

`related` 与当前文献某种程度关联的文献的关键词的分离列表³⁴。注意: 关键词³⁵的顺序很重要。来自多个关联文献的数据是按该域中关键词的顺序打印。

³³这里 `related data` 用关联而不是相关, 其它地方用到相关的应一并改过来

³⁴这里 `separated list` 用分离列表有没有更好的说法

³⁵这里的 `key` 关键词应该是引用关键词, 即 `bibtex` 键

- relatedtype** The type of relationship. This serves three purposes. If the value of this field resolves to a localization string identifier, then the resulting localized string is printed before the data from the related entries. Secondly, if there is a macro called `related:<relatedtype>`, this is used to format the data from the related entries. If no such macro exists, then the macro `related:default` is used. Lastly, if there is a format named `related:<relatedtype>`, then it is used to format both the localized string and related entry data. If there is no related type specific format, the `related` format is used.
- relatedstring** If an entry contains this field, then if value of the field resolves to a localization string identifier, the localization key value specified is printed before data from the related entries. If the field does not specify a localization key, its value is printed literally. If both `relatedtype` and `relatedstring` are present in an entry, `relatedstring` is used for the pre-data string (but `relatedtype` is still used to determine the macro and format to use when printing the data).
- relatedoptions** A list of per-entry options to set on the related entry (actually on the clone of the related entry which is used as a data source—the actual related entry is not modified because it might be cited directly itself).

The related entry feature is enabled by default by the package option `related` from § 3.1.2.1. The related information entry data from the related entries is included via a `\usebibmacro{related}` call. Standard styles call this macro towards the end of each driver. Style authors should ensure the existence of (or take note of existing) localization strings which are useful as values for the `relatedtype` field, such as `translationof` or perhaps `translatedas`. A plural variant can be identified with the localization key `<relatedtype>s`. This key's corresponding string is printed whenever more than one entry is specified in `related`. Bibliography macros and formatting directives for printing entries related by `<relatedtype>` should be defined using the name `related:<relatedtype>`. The file `biblatex.def` contains macros and formats for some common relation types which can be used as templates. In particular, the `\entrydata*` command is essential in such macros in order to make the data of the related entries available. Examples of entries using this feature can be found in the Biblatex distribution examples file `biblatex-examples.bib`. There are some specific formatting macros for this feature which control delimiters and separators in related entry information, see § 4.10.1.

4.5.2 数据动态修改 Dynamic Modification of Data

Bibliographic data sources which are automatically generated or which you have no control over can be a problem if you need to edit them in some way. For this reason, Biber has the ability to modify data as it is read so that you can apply modifications to the source data stream without actually changing it. The modification can be defined in Biber's config file (see Biber docs), or via Biblatex macros in which case you can apply the modification only for specific documents, styles or globally.

Source mappings can be defined at different “levels” which are applied in a defined order. See the Biblatex manual regarding these macros:

user-level maps defined with `\DeclareSourcemap`→
 user-level maps defined in the Biber config file (see Biber docs)→
 style-level maps defined with `\DeclareStyleSourcemap`→
 driver-level maps defined with `\DeclareDriverSourcemap`

`\DeclareSourcemap{⟨specification⟩}`

Biber only

Defines source data modification (mapping) rules which can be used to perform any combination of the following tasks:

- Map data source entrytypes to different entrytypes
- Map datasource fields to different fields
- Add new fields to an entry
- Remove fields from an entry
- Modify the contents of a field using standard Perl regular expression match and replace
- Restrict any of the above operations to entries coming from particular datasources which you defined in `\addresource` macros
- Restrict any of the above operations to entries only of a certain entrytype

The `⟨specification⟩` is an undelimited list of `\maps` directives which specify containers for mappings rules applying to a particular data source type (§ 3.6.1). Spaces, tabs, and line endings may be used freely to visually arrange the `⟨specification⟩`. Blank lines are not permissible. This command may only be used in the preamble and may only be used once—subsequent uses will overwrite earlier definitions.

`\maps{⟨elements⟩}`

Contains an ordered set of `\map` elements each of which is a logically related set of mapping steps to apply to the data source.

`datatype=bibtex, biblatexml, ris, zoterordfxml, endnotexml` **default:** `bibtex`

Data source type to which the contained `\map` directives apply (§ 3.6.1).

`overwrite=true, false` **default:** `false`

Specify whether a mapping rule is allowed to overwrite already existing data in an entry. If this option is not specified, the default is `false`. The short form `overwrite` is equivalent to `overwrite=true`.

`\map{⟨restrictions,steps⟩}`

A container for an ordered set of map `\steps`, optionally restricted to particular entrytypes or data sources. This is a grouping element to allow a set of mapping steps to apply only to specific entrytypes or data sources. Mapping steps must always be contained within a `\map` element.

`overwrite=true, false`

As the same option on the parent `\maps` element. This option allows an override on a per-map group basis. If this option is not specified, the default is the parent `\maps` element option value. The short form `overwrite` is equivalent to `overwrite=true`.

`\perdatasource{⟨datasource⟩}`

Restricts all `\steps` in this `\map` element to entries from the named `⟨datasource⟩`. The `⟨datasource⟩` name should be exactly as given in a `\addresource` macro defining a data source for the document. Multiple `\perdatasource` restrictions are allowed within a `\map` element.

`\pertype{⟨entrytype⟩}`

Restricts all `\steps` in this `\map` element to entries of the named `⟨entrytype⟩`. Multiple `\pertype` restrictions are allowed within a `\map` element.

`\pernottype{⟨entrytype⟩}`

Restricts all `\steps` in this `\map` element to entries not of the named `⟨entrytype⟩`. Multiple `\pernottype` restrictions are allowed within a `\map` element.

`\step`

A mapping step. Each step is applied sequentially to every relevant entry where ‘relevant’ means those entries which correspond to the data source type, entrytype and data source name restrictions mentioned above. Each step is applied to the entry as it appears after the application of all previous steps. The mapping performed by the step is determined by the following options:

`typesource=⟨entrytype⟩`

`typetarget=⟨entrytype⟩`

`fieldsource=⟨entryfield⟩`

`fieldtarget=⟨entryfield⟩`

`match=⟨regexp⟩`

`notmatch=⟨regexp⟩`

`replace=⟨regexp⟩`

`fieldset=⟨entryfield⟩`

<code>fieldvalue=<string></code>	
<code>entryclone=<string></code>	
<code>entrynull=true, false</code>	default: false
<code>append=true, false</code>	default: false
<code>final=true, false</code>	default: false
<code>null=true, false</code>	default: false
<code>origfield=true, false</code>	default: false
<code>origfieldval=true, false</code>	default: false
<code>origentrytype=true, false</code>	default: false

For all boolean \step options, the short form option is equivalent to option=true. The following rules for a mapping step apply:

- If `entrynull` is set, processing of the \map immediately terminates and the current entry is not created. It is as if it did not exist in the datasource. Obviously, you should select the entries which you want to apply this to using prior mapping steps.
- If `entryclone` is set, a clone of the entry is created with an entry key prefixed by `prefix`. Obviously this may cause labelling problems in author/year styles etc. and should be used with care. It is mainly intended for numbering issues with numeric styles and multiple bibliographies where the same entries occur in more than one bibliography.
- Change the `typesource <entrytype>` to the `typetarget <entrytype>`, if defined. If `final` is true then if the `<entrytype>` of the entry is not `typesource`, processing of the parent \map immediately terminates.
- Change the `fieldsource <entryfield>` to `fieldtarget`, if defined. If `final` is true then if there is no `fieldsource <entryfield>` in the entry, processing of the parent \map immediately terminates.
- If `match` is defined but `replace` is not, only apply the step if the `fieldsource <entryfield>` matches the `match` regular expression (logic is reversed if you use `notmatch` instead)³⁶. You may use capture parenthesis as usual and refer to these (\$1...\$9) in later `fieldvalue` specifications. This allows you to pull out parts of some fields and put these parts in other fields.
- Perform a regular expression match and replace on the value of the `fieldsource <entryfield>` if `match` and `replace` are defined.
- If `fieldset` is defined, then its value is `<entryfield>` which will be set to a value specified by further options. If `overwrite` is false for this step and the field to set already exists then the map step is ignored. If `final` is also true for this step,

³⁶Regular expressions are full Perl 5.16 regular expressions. This means you may need to deal with special characters, see examples.

then processing of the parent map stops at this point. If `append` is true, then the value to set is appended to the current value of `<entryfield>`. The value to set is specified by a mandatory one and only one of the following options:

- `fieldvalue` — The fieldset `<entryfield>` is set to the fieldvalue `<string>`
- `null` — The fieldset `<entryfield>` is ignored, as if it did not exist in the datasource
- `origentrytype` — The fieldset `<entryfield>` is set to the most recently mentioned typesource `<entrytype>` name
- `origfield` — The fieldset `<entryfield>` is set to the most recently mentioned fieldsource `<entryfield>` name
- `origfieldval` — The fieldset `<entryfield>` is set to the most recently mentioned fieldsource value

With BibTeX and RIS datasources, you may specify the pseudo-field `entrykey` for fieldsource which is the citation key of the entry. Naturally, this ‘field’ cannot be changed (used as `fieldset`, `fieldtarget` or changed using `replace`).

`\DeclareStyleSourcemap{<specification>}`

Biber only

This command sets the source mappings used by a style. Such mappings are conceptually separate from user mappings defined with `\DeclareSourcemap` and are applied directly after user maps. The syntax is identical to `\DeclareSourcemap`. This command is provided for style authors so that any maps defined for the style do not interfere with user maps or the default driver maps defined with `\DeclareDriverSourcemap`. This command is for use in style files and can only be used once—subsequent uses will overwrite earlier definitions.

`\DeclareDriverSourcemap[<datatype=driver>]{<specification>}`

Biber only

This command sets the driver default source mappings for the specified `<driver>`. Such mappings are conceptually separate from user mappings defined with `\DeclareSourcemap` and style mapping defined with `\DeclareStyleSourcemap`. They consist of mappings which are part of the driver setup. Users should not normally need to change these. Driver default mapping are applied after user mappings (`\DeclareSourcemap`) and style mappings (`\DeclareStyleSourcemap`). These defaults are described in Appendix § A. The `<specification>` is identical to that for `\DeclareSourcemap` but without the `\maps` elements: the `<specification>` is just a list of `\map` elements since each `\DeclareDriverSourcemap` only applies to one datatype driver. See the default definitions in Appendix § A for examples.

Here are some data source mapping examples:

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \perdatasource{example1.bib}
      \perdatasource{example2.bib}
      \step[fieldset=keywords, fieldvalue={keyw1, keyw2}]
      \step[fieldsource=entrykey]
      \step[fieldset=note, origfieldval]
    }
  }
}

```

This would add a keywords field with value ‘keyw1, keyw2’ and set the note field to the entry key to all entries which are found in either the examples1.bib or examples2.bib files.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=title]
      \step[fieldset=note, origfieldval]
    }
  }
}

```

Copy the title field to the note field unless the note field already exists.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[typesource=chat, typetarget=customa, final]
      \step[fieldset=type, origentrytype]
    }
  }
}

```

Any chat entrytypes would become customa entrytypes and would automatically have a type field set to ‘chat’ unless the type field already exists in the entry (because overwrite is false by default). This mapping applies only to entries of type @chat since the first step has final set and so if the typesource does not match the entry entrytype, processing of this \map immediately terminates.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{

```

```

\perdatasource{examples.bib}
\pertype{article}
\pertype{book}
\step[fieldset=abstract, null]
\step[fieldset=note, fieldvalue={Auto-created this field}]
}
}
}

```

Any entries of entrytype @article or @book from the examples.bib datasource would have their abstract fields removed and a note field added with value ‘Auto-created this field’.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldset=abstract, null]
      \step[fieldsource=conductor, fieldtarget=namea]
      \step[fieldsource=gps, fieldtarget=usera]
    }
  }
}

```

This removes abstract fields from any entry, changes conductor fields to namea fields and changes gps fields to usera fields.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=pubmedid, fieldtarget=eprint, final]
      \step[fieldset=eprinttype, origfield]
      \step[fieldset=userd, fieldvalue={Some string of things}]
    }
  }
}

```

Applies only to entries with pubmed fields and maps pubmedid fields to eprint fields, sets the eprinttype field to ‘pubmedid’ and also sets the userd field to the string ‘Some string of things’.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=series,
        match=\regexp{\A\d*(.+)},

```

```

        replace=\regexp{\L$1}]
    }
}

```

Here, the contents of the `series` field have leading numbers stripped and the remainder of the contents lowercased. Since regular expressions usually contain all sort of special characters, it is best to enclose them in the provided `\regexp` macro as shown—this will pass the expression through to Biber correctly.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=maintitle,
            match=\regexp{Collected\s+Works.+Freud},
            final]
      \step[fieldset=keywords, fieldvalue=freud]
    }
  }
}

```

Here, if for an entry, the `maintitle` field matches a particular regular expression, we set a special keyword so we can, for example, make a references section just for certain items.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=lista, match=\regexp{regexp}, final]
      \step[fieldset=lista, null]
    }
  }
}

```

If an entry has a `lista` field which matches regular expression ‘`regexp`’, then it is removed.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite=false]{
      \step[fieldsource=author]
      \step[fieldset=editor, origfieldval, final]
      \step[fieldsource=editor, match=\regexp{\A(.+?)\s+and.*}, replace={$1}]
    }
  }
}

```

For any entry with an author field, try to set sortname to the same as author. If this fails because sortname already exists, stop, otherwise truncate sortname to just the first name in the name list.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=author,
            match={Smith, Bill},
            replace={Smith, William}]
      \step[fieldsource=author,
            match={Jones, Baz},
            replace={Jones, Barry}]
    }
  }
}
```

Here, we use multiple match/replace for the same field to regularise some inconstant name variants. Bear in mind that `\step` processing within a `map` element is sequential and so the changes from a previous `\steps` are already committed. Note that we don't need the `\regex` macro to protect the regular expressions in this example as they contain no characters which need special escaping. Please note that due to the difficulty of protecting regular expressions in \LaTeX , there should be no literal spaces in the argument to `\regex`. Please use escape code equivalents if spaces are needed. For example, this example, if using `\regex`, should be:

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=author,
            match=\regex{Smith,\s+Bill},
            replace=\regex{Smith,\x20William}]
      \step[fieldsource=author,
            match=\regex{Jones,\s+Baz},
            replace=\regex{Jones,\x20Barry}]
    }
  }
}
```

Here, we have used the hexadecimal escape sequence `'\x20'` in place of literal spaces in the replacement strings.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
```

```

\map[overwrite]{
  \step[fieldsource=author, match={Doe,}, final]
  \step[fieldset=shortauthor, origfieldval]
  \step[fieldset=sortname, origfieldval]
  \step[fieldsource=shortauthor,
    match=\regexp{Doe,\s*(?:\.|ohn) (?:[-]*) (?:P\.|Paul)*},
    replace={Doe, John Paul}]
  \step[fieldsource=sortname,
    match=\regexp{Doe,\s*(?:\.|ohn) (?:[-]*) (?:P\.|Paul)*},
    replace={Doe, John Paul}]
}
}
}

```

Only applies to entries with an author field matching ‘Doe.’. First the author field is copied to both the shortauthor and sortname fields, overwriting them if they already exist. Then, these two new fields are modified to canonicalise a particular name, which presumably has some variants in the data source.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite]{
      \step[fieldsource=verba, final]
      \step[fieldset=verbb, fieldvalue=/, append]
      \step[fieldset=verbb, origfieldval, append]
      \step[fieldsource=verbb, final]
      \step[fieldset=verbc, fieldvalue=/, append]
      \step[fieldset=verbc, origfieldval, append]
    }
  }
}

```

This example demonstrates the sequential nature of the step processing and the append option. If an entry has a verba field then first, a forward slash is appended to the verbb field. Then, the contents of verba are appended to the verbb field. A slash is then appended to the verbc field and the contents of verbb are appended to the verbc field.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite]{
      \step[fieldset=autourl, fieldvalue={http://scholar.google.com/scholar?}
      \step[fieldsource=title]
      \step[fieldset=autourl, origfieldval, append]

```

```

\step[fieldset=autourl, fieldvalue={"author:}, append]
\step[fieldsource=author, match=\regexp{\A([^\,]+)\s*,}]
\step[fieldset=autourl, fieldvalue={$1}, append]
\step[fieldset=autourl, fieldvalue={\as_ylo=}, append]
\step[fieldsource=year]
\step[fieldset=autourl, origfieldval, append]
\step[fieldset=autourl, fieldvalue={\as_yhi=}, append]
\step[fieldset=autourl, origfieldval, append]
}
}
}

```

This example assumes you have created a field called `autourl` using the `datamodel` macros from § 4.5.3 in order to hold, for example, a Google Scholar query URL auto-created from elements of the entry. The example progressively extracts information from the entry, constructing the URL as it goes. It demonstrates that it is possible to refer to parenthetical matches from the most recent match in any following `fieldvalue` which allows extracting the lastname from the `author`, assuming a ‘last, first’ format. The resulting field could then be used as a hyperlink from, for example, the title of the work in the bibliography.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=title, match={A Title}, final]
      \step[entrynull]
    }
  }
}

```

Any entry with a `title` field matching ‘A Title’ will be completely ignored.

```

\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \pernottype{book}
      \pernottype{article}
      \step[entrynull]
    }
  }
}

```

Any entry which is not a `@book` or `@article` will be ignored.

```

\DeclareSourcemap{

```

```

\maps[datatype=bibtex]{
  \map{
    \perdatasource{biblatex-examples.bib}
    \step[entryclone={rel-}]
  }
}

```

Here, a clone of an entry from the specified data source will be created. The entry key of the clone will be the same as the original but prefixed by the value of the `entryclone` parameter. The cloned entry would still need to be cited in the document using its new entry key. This type of mapping step should be used with care as it may produce labelling problems in authoryear styles which use, for example, `extrayear`. One use case is for numeric styles which contain multiple bibliographies containing the same entry. In this case, you may need different bibliography number label for the same entry and this is very tricky when there is only one entry which needs different labels. Creating clones with different entry keys solves this problem.

4.5.3 数据模型规范 Data Model Specification

Biber only

The data model which Biblatex uses consists of four main elements:

- Specification of valid Entrytypes
- Specification of valid Fields along with their type, datatype and any special flags
- Specification of which Fields are valid in which Entrytypes
- Specification of constraints which can be used to validate data against the data model

The default data model is defined in the core Biblatex file `blx-dm.def` using the macros described in this section. The default data model is described in detail in § 2. The data model is used internally by Biblatex and also by the backends. The data model for the BibTeX backend is hard-coded and cannot be changed. Changing the data model is only supported for the Biber backend. In practice, changing the data model means that you can define the entrytypes and fields for your datasources and validate your data against the data model. Naturally, this is not much use unless your style supports any new entrytypes or fields and it raises issues of portability between styles (although this can be mitigated by using the dynamic data modification functionality described in § 4.5.2).

Validation against the data model means that after mapping your data sources into the data model, Biber (using its `--validate_datamodel` option) can check:

- Whether all entrytypes are valid entrytypes

- Whether all fields are valid fields for their entrytype
- Whether the fields obey various constraints on their format which you specify

Redefining the data model can be done in several places. Style authors can create a `.dbx` file which contains the data model macros required and this will be loaded automatically when using the Biblatex package `style` option by looking for a file named after the style with a `.dbx` extension (just like the `.cbx` and `.bbx` files for a style). If the `style` option is not used but rather the `citestyle` and `bibstyle` options, then the package will try to load `.dbx` files called `'citestyle'.dbx` and `'bibstyle'.dbx`. Alternatively, the name of the data model file can be different from any of the style option names by specifying the name (without `.dbx` extension) to the package `datamodel` option. After loading the style data model file, Biblatex then loads, if present, a users `biblatex-dm.cfg` which should be put somewhere Biblatex can find it, just like the main configuration file `biblatex.cfg`. To summarise, the data model is determined by adding to the data model from each of these locations, in order:

```
blx-dm.def→
'datamodel option'.dbx →
'style option'.dbx →
'citestyle option'.dbx and 'bibstyle option'.dbx →
biblatex-dm.cfg
```

It is not possible to add to a loaded data model by using the macros below in your preamble as the preamble is read after Biblatex has defined critical internal macros based on the data model. If any data model macro is used in a document, it will be ignored and a warning will be generated. The data model is defined using the following macros:

`\DeclareDatamodelEntrytypes` [*options*] {*entrytypes*}

Biber only

Declares the comma-separated list of *entrytypes* to be valid entrytypes in the data model. As usual in TeX csv lists, make sure each element is immediately followed by a comma or the closing brace—no extraneous whitespace.

`skipout`=true, false default: false

This entrytype is not output to the `.bbl`. Typically used for special entrytypes which are processed and consumed by the backend such as `@xdata`.

`\DeclareDatamodelFields` [*options*] {*fields*}

Biber only

Declares the comma-separated list of *fields* to be valid fields in the data model with associated comma-separated *options*. The *type* and *datatype* options are mandatory. All valid *options* are:

`type=⟨field type⟩`

Set the type of the field as described in § 2.2.1. Is typically ‘field’ or ‘list’.

`format=⟨field format⟩`

Any special format of the field. Normally unspecified but can take the value ‘xsv’ which tells Bibex that this field has a separated values format. The exact separator can be controlled with the Bibex option `xsvsep` and defaults to the expected comma surrounded by optional whitespace.

`datatype=⟨field datatype⟩`

Set the datatype of the field as described in § 2.2.1. For example, ‘name’ or ‘literal’.

`nullok=true, false` default: false

The field is allowed to be defined but empty.

`skipout=true, false` default: false

The field is not output to the .bbl and is therefore not present during Biblatex style processing. As usual in TeX csv lists, make sure each element is immediately followed by a comma or the closing brace—no extraneous whitespace.

`label=true, false` default: false

The field can be used as a label in a bibliography or bibliography list. Specifying this causes Biblatex to create several helper macros for the field so that there are some internal lengths and headings etc. defined.

`\DeclareDatamodelEntryfields [⟨entrytypes⟩] {⟨fields⟩}`

Biber only

Declares that the comma-separated list of ⟨fields⟩ is valid for the comma-separated list of ⟨entrytypes⟩. If ⟨entrytypes⟩ is not given, the fields are valid for all entrytypes. As usual in TeX csv lists, make sure each element is immediately followed by a comma or the closing brace—no extraneous whitespace.

`\DeclareDatamodelConstraints [⟨entrytypes⟩] {⟨specification⟩}`

Biber only

If a comma-separated list of ⟨entrytypes⟩ is given, the constraints apply only to those entrytypes. The ⟨specification⟩ is an undelimited list of `\constraint` directives which specify a constraint. Spaces, tabs, and line endings may be used freely to visually arrange the ⟨specification⟩. Blank lines are not permissible.

`\constraint [⟨type=constrainttype⟩] {⟨elements⟩}`

Specifies a constraint of type ⟨constrainttype⟩. Valid constraint types are:

`type=data, mandatory, conditional`

Constraints of type ‘data’ put restrictions on the value of a field. Constraints of type ‘mandatory’ specify which fields or combinations of fields an entrytype should have. Constraints of type ‘conditional’ allow more sophisticated conditional and quantified field constraints.

`datatype=integer, isbn, issn, ismn, date, pattern`

For constraints of type $\langle data \rangle$, constrain field values to be the given datatype.

`rangemin= $\langle num \rangle$`

For constraints of $\langle type \rangle$ ‘data’ and $\langle datatype \rangle$ ‘integer’, constrain field values to be at least $\langle num \rangle$.

`rangemax= $\langle num \rangle$`

For constraints of $\langle type \rangle$ ‘data’ and $\langle datatype \rangle$ ‘integer’, constrain field values to be at most $\langle num \rangle$.

`pattern= $\langle patt \rangle$`

For constraints of $\langle type \rangle$ ‘data’ and $\langle datatype \rangle$ ‘pattern’, constrain field values to match regular expression pattern $\langle patt \rangle$. It is best to wrap any regular expression in the macro `\regexp`, see § 4.5.2.

A `\constraint` macro may contain any of the following:

`\constraintfieldsor{ $\langle fields \rangle$ }`

For constraints of $\langle type \rangle$ ‘mandatory’, specifies that an entry must contain a boolean OR of the `\constraintfields`.

`\constraintfieldsxor{ $\langle fields \rangle$ }`

For constraints of $\langle type \rangle$ ‘mandatory’, specifies that an entry must contain a boolean XOR of the `\constraintfields`.

`\antecedent[$\langle quantifier=quantspec \rangle$]{ $\langle fields \rangle$ }`

For constraints of $\langle type \rangle$ ‘conditional’, specifies a quantified set of `\constraintfields` which must be satisfied before the `\consequent` of the constraint is checked. $\langle quantspec \rangle$ should have one of the following values:

`quantifier=all, one, none`

Specifies how many of the `\constraintfield`’s inside the `\antecedent` have to be present to satisfy the antecedent of the conditional constraint.

`\consequent[$\langle quantifier=quantspec \rangle$]{ $\langle fields \rangle$ }`

For constraints of $\langle type \rangle$ ‘conditional’, specifies a quantified set of `\constraintfields` which must be satisfied if the preceding `\antecedent` of the constraint was satisfied. $\langle quantspec \rangle$ should have one of the following values:

`quantifier=all, one, none`

Specifies how many of the `\constraintfield`’s inside the `\consequent` have to be present to satisfy the consequent of the conditional constraint.

`\constraintfield{field}`

For constraints of *<type>* ‘data’, the constraint applies to this *<field>*. For constraints of *<type>* ‘mandatory’, the entry must contain this *<field>*.

The data model declaration macros may be used multiple times as they append to the previous definitions. In order to replace, change or remove existing definitions (such as the default model which is loaded with Biblatex), you should reset (clear) the current definition and then set what you want using the following macros. Typically, these macros will be the first things in any `biblatex-dm.cfg` file:

`\ResetDatamodelEntrytypes`

Clear all data model entrytype information.

`\ResetDatamodelFields`

Clear all data model field information.

`\ResetDatamodelEntryfields`

Clear all data model fields for entrytypes information.

`\ResetDatamodelConstraints`

Clear all data model fields Constraints information.

Here is an example of a simple data model. Refer to the core Biblatex file `blx-dm.def` for the default data model specification.

```
\ResetDatamodelEntrytypes
\ResetDatamodelFields
\ResetDatamodelEntryfields
\ResetDatamodelConstraints
```

```
\DeclareDatamodelEntrytypes{entrytype1, entrytype2}
```

```
\DeclareDatamodelFields[type=field, datatype=literal]{field1, field2, field3, field4}
```

```
\DeclareDatamodelEntryfields{field1}
```

```
\DeclareDatamodelEntryfields[entrytype1]{field2, field3}
```

```
\DeclareDatamodelEntryfields[entrytype2]{field2, field3, field4}
```

```
\DeclareDatamodelConstraints[entrytype1]{
```

```
  \constraint[type=data, datatype=integer, rangemin=3, rangemax=10]{
```

```
    \constraintfield{field1}
```

```
  }
```

```
  \constraint[type=mandatory]{
```

```

\constraintfield{field1}
\constraintfieldsxor{
  \constraintfield{field2}
  \constraintfield{field3}
}
}
}
\DeclareDataModelConstraints{
  \constraint[type=conditional]{
    \antecedent[quantifier=none]{
      \constraintfield{field2}
    }
    \consequent[quantifier=all]{
      \constraintfield{field3}
      \constraintfield{field4}
    }
  }
}
}

```

This model specifies:

- Clear the default data model completely
- Two valid entry types @entrytype1 and @entrytype2
- Four valid literal field fields
- field1 is valid for all entrytypes
- field2 and field3 are valid for entrytype1
- field2, field3 and field4 are valid for @entrytype2
- For @entrytype1:
 - field1 must be an integer between 3 and 10
 - field1 must be present
 - One and only one of field2 or field3 must be present
- For any entrytype, if field2 is not present, field3 and field4 must be present

4.5.4 标签 Labels

字母顺序制样式使用一个标签来区分参考文献条目。这个标签由条目的内容使用一个描述怎么构建标签的模板构建。该模板可以全局自定义或者分条目类型定义。标签的自定义需要用 Biber 后端而不能用其它后端。

`\DeclareLabelalphaTemplate[⟨entrytype, ...⟩]{⟨specification⟩}`

Biber only

Defines the alphabetic label template for the given entrytypes. If no entrytypes are specified in the first argument, then the global label template is defined. The *⟨specification⟩* is an undelimited list of `\labelelement` directives which specify the elements used to build the label. Spaces, tabs, and line endings may be used freely to visually arrange the *⟨specification⟩*. Blank lines are not permissible. This command may only be used in the preamble.

`\labelelement{⟨elements⟩}`

Specifies the elements used to build the label. The *⟨elements⟩* are an undelimited list of `\field` or `\literal` commands which are evaluated in the order in which they are given. The first `\field` or `\literal` which expands to a non-empty string is used as the `\labelelement` expansion and the next `\labelelement`, if any, is then processed.

`\field[⟨options⟩]{⟨field⟩}`

If *⟨field⟩* is non-empty, use it as the current label `\labelelement`, subject to the options below. Useful values for *⟨field⟩* are typically the name list type fields, date fields, and title fields. You may also use the ‘citekey’ pseudo-field to specify the citation key as part of the label. Name list fields are treated specially and the options which take substrings of the *⟨field⟩* to use in the `\labelelement` are applied to the lastname of every name in a name list (see below).

`final=true, false` default: false

This option marks a `\field` directive as the final one in the *⟨specification⟩*. If the *⟨field⟩* is non-empty, then this field is used for the label and the remainder of the *⟨specification⟩* will be ignored. The short form `final` is equivalent to `final=true`.

`lowercase=true, false` default: false

Forces the label part derived from the field to lowercase. By default, the case is taken from the field source and not modified.

`strwidth=⟨integer⟩` default: 1

The number of characters of the *⟨field⟩* to use. For name list fields, if `useprefix=true`, the first character of the name prefix is prepended to the substring.

`strside=left, right` default: left

The side of the string from which to take the `strwidth` number of characters.

`padside=left, right` default: right

Side to pad the label part when using the `padchar` option. Only for use with fixed-width label strings (`strwidth`).

`padchar=<character>`

If present, pads the label part on the `padside` side with the specified character to the length of `strwidth`. Only for use with fixed-width label strings (`strwidth`).

`uppercase=true, false` default: false

Forces the label part derived from the field to uppercase. By default, the case is taken from the field source and not modified.

`varwidth=true, false` default: false

Use a variable width, left-side substring of characters from the *<field>* (each lastname in name list fields). The length of the string is determined by the minimum length needed to disambiguate the substring from all other *<field>* elements in the same position in the label. For name list fields, this means that each name substring is disambiguated from all other name substrings which occur in the same position in the name list (see examples below). This option overrides `strwidth` if both are used. The short form `varwidth` is equivalent to `varwidth=true`. For name list fields, if `useprefix=true`, the first character of the name prefix is prepended to the substring.

`varwidthnorm=true, false` default: false

As `varwidth` but will force the disambiguated substrings for the *<field>* to be the same length as the longest disambiguated substring. This can be used to regularise the format of the labels if desired. This option overrides `strwidth` if both are used. The short form `varwidthnorm` is equivalent to `varwidthnorm=true`.

`varwidthlist=true, false` default: false

Alternative method of automatic label disambiguation where the field as a whole is disambiguated from all other fields in the same label position. For non-name list fields, this is equivalent to `varwidth`. For name list fields, names in a name list are not disambiguated from other names in the same position in their name lists but instead the entire name list is disambiguated as a whole from other name lists (see examples below). This option overrides `strwidth` if both are used. The short form `varwidthlist` is equivalent to `varwidthlist=true`. For name list fields, if `useprefix=true`, the first character of the name prefix is prepended to the substring.

`strwidthmax=<integer>`

When using `varwidth`, this option sets a limit (in number of characters) on the length of variable width substrings. This option can be used to regularise the label.

`strfixedcount=<integer>` default: 1

When using `varwidthnorm`, there must be at least `strfixedcount` disambiguated substrings with the same, maximal length to trigger the forcing of all disambiguated substrings to this same maximal length.

`compound=true, false` default: false

For static (non-`varwidth`) disambiguation, treat lastname name components separated by whitespace or hyphens (compound last names) as separate names for label gener-

ation. This means that when forming a label out of, for example the surname ‘Balam Forsyth’ with a 1 character, left-side substring, this name would give ‘BF’ with `compound=true` and ‘B’ with `compound=false`. The short form `compound` is equivalent to `compound=true`.

`ifnames=<integer>`

Only use this `\field` specification if it is a name list field with `ifnames` names in it. This allows a `\labelelement` to be conditionalised on name length (see below).

`names=<integer>`

By default, for name list fields, the number of names considered when building the label obeys the `maxalphanames/minalphanames` truncation. This option can be used to override this with an explicit upper limit of how many names to consider. This is useful if you have some very long name lists which might result in a very long label.

`form=original, translated, romanised, uniform` default: `original`

Specifies the script variant of a field, if the datasource you are using supports this.

`lang='language specifier'` default: `none`

Specifies the `babel/polyglossia` language variant of a field if the datasource you are using supports this. The default, if no `<lang>` option is set, this means to use the field variant in the datasource which has no explicit language defined.

`\literal{<characters>}`

Insert the literal `<characters>` into the label at this point.

Note that the template for labels can be defined per-type and you should be aware of this when using the automatically disambiguated label functionality. Disambiguation is not per-type as this might lead to ambiguity due to different label formats for different types being isolated from each others disambiguation process. Normally, you will want to use very different label formats for different types to make the type obvious by the label.

Here are some examples. The default global Biblatex alphabetic label template is defined below. Firstly, `shorthand` has `final=true` and so if there is a `shorthand` field, it is used as the label and nothing more of the template is considered. Next, the `label` field is used as the first label element if it exists. Otherwise, if there is only one name (`ifnames=1`) in the `labelname` list, then three characters from the left side of the `lastname` in the `labelname` are used as the first label element. If the `labelname` has more than one name in it, one character from the left side of each `lastname` is used as the first label element. The second label element consists of 2 characters from the right side of the `year` field.

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[final]{shorthand}
```



```

\field{label}
\field[strwidth=3, strside=left, ifnames=1]{labelname}
\field[strwidth=1, strside=left]{labelname}
}
\labelelement{
\field[strwidth=2, strside=right]{year}
}
}

```

To get an idea of how the label automatic disambiguation works, consider the following author lists:

```

Agassi, Chang, Laver    (2000)
Agassi, Connors, Lendl (2001)
Agassi, Courier, Laver  (2002)
Borg, Connors, Edberg   (2003)
Borg, Connors, Emerson (2004)

```

Assuming a template declaration such as:

```

\DeclareLabelalphaTemplate{
\labelelement{
\field[varwidth]{labelname}
}
}

```

Then the labels would be:

```

Agassi, Chang, Laver    [AChLa]
Agassi, Connors, Lendl  [AConLe]
Agassi, Courier, Laver  [ACouLa]
Borg, Connors, Edberg   [BConEd]
Borg, Connors, Emerson [BConEm]

```

With normalised variable width labels defined:

```

\DeclareLabelalphaTemplate{
\labelelement{
\field[varwidthnorm]{labelname}
}
}

```

You would get the following as the substrings of names in each position are extended to the length of the longest substring in that same position:

Agassi, Chang, Laver	[AChLa]
Agassi, Connors, Lendl	[AConLe]
Agassi, Courier, Laver	[ACouLa]
Borg, Connors, Edberg	[BConEd]
Borg, Connors, Emerson	[BConEm]

With a restriction to two characters for the name components of the label element defined like this:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthnorm, strwidthmax=2]{labelname}
  }
}
```

This would be the result (note that the individual lastname label parts are no longer unambiguous):

Agassi, Chang, Laver	[AChLa]
Agassi, Connors, Lendl	[ACoLe]
Agassi, Courier, Laver	[ACoLa]
Borg, Connors, Edberg	[BCoEd]
Borg, Connors, Emerson	[BCoEm]

Alternatively, you could choose to disambiguate the name lists as a whole with:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthlist]{labelname}
  }
}
```

Which would result in:

Agassi, Chang, Laver	[AChL]
Agassi, Connors, Lendl	[ACoL]
Agassi, Courier, Laver	[ACL]
Borg, Connors, Edberg	[BCEd]
Borg, Connors, Emerson	[BCE]

Perhaps you only want to consider at most two names for label generation but disambiguate at the whole name list level:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthlist, names=2]{labelname}
  }
}
```

Which would result in:

```
Agassi, Chang, Laver      [ACh+]
Agassi, Connors, Lendl    [ACo+]
Agassi, Courier, Laver    [AC+]
Borg, Connors, Edberg     [BC+a]
Borg, Connors, Emerson    [BC+b]
```

In this last example, you can see `\labelalphaothers` has been appended to show that there are more names. The last two labels now require disambiguating with `\extraalpha` as there is no way of disambiguating this label name list with only two names.

Finally, here is an example using multiple label elements:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthlist]{labelname}
  }
  \labelelement{
    \literal{-}
  }
  \labelelement{
    \field[strwidth=3, strside=right]{labelyear}
  }
}
```

Which would result in:

```
Agassi, Chang, Laver      [AChL-000]
Agassi, Connors, Lendl    [AConL-001]
Agassi, Courier, Laver    [ACouL-002]
Borg, Connors, Edberg     [BCEd-003]
Borg, Connors, Emerson    [BCEm-004]
```

Here is another rather contrived example showing that you don't need to specially quote LaTeX special characters (apart from '%', obviously) when specifying padding characters and literals:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \literal{>}
  }
  \labelelement{
    \literal{\%}
  }
}
```

```

\labelement{
  \field[strwidth=4, padchar=_]{labelname}
}
\labelement{
  \field[strwidth=3, padchar=&, padside=left]{title}
}
\labelement{
  \field[strwidth=2, strside=right]{year}
}
}

```

which given:

```

@Book{test,
  author   = {XXX YY},
  title    = {T},
  year     = {2007},
}

```

would resulting a label looking like this:

```
[>%YY__&&T07]
```

4.5.5 排序 Sorting

In addition to the predefined sorting schemes discussed in § 3.5, it is possible to define new ones or modify the default definitions. The sorting process may be customized further by excluding certain fields from sorting on a per-type basis and by automatically populating the `presort` field on a per-type basis. Note that custom sorting schemes require Biber. They will not work with any other backend.

`\DeclareSortingScheme`[$\langle options \rangle$]{ $\langle name \rangle$ }{ $\langle specification \rangle$ }

Biber only

Defines the sorting scheme $\langle name \rangle$. The $\langle name \rangle$ is the identifier passed to the sorting option (§ 3.1.2.1) when selecting the sorting scheme. The `\DeclareSortingScheme` command supports the following optional arguments:

`locale`= $\langle locale \rangle$

The locale for the sorting scheme which then overrides the global sorting locale in the `sortlocale` option discussed in § 3.1.2.1.

The $\langle specification \rangle$ is an undelimited list of `\sort` directives which specify the elements to be considered in the sorting process. Spaces, tabs, and line endings may be used freely to visually arrange the $\langle specification \rangle$. Blank lines are not permissible. This command may only be used in the preamble.

`\sort{⟨elements⟩}`

Specifies the elements considered in the sorting process. The *⟨elements⟩* are an undelimited list of `\field`, `\literal`, and `\citeorder` commands which are evaluated in the order in which they are given. If an element is defined, it is added to the sort key and the sorting routine skips to the next `\sort` directive. If it is undefined, the next element is evaluated. Since literal strings are always defined, any `\literal` commands should be the sole or the last element in a `\sort` directive. The `\sort` command supports the following optional arguments:

`locale=⟨locale⟩`

Override the locale used for sorting at the level of a particular set of sorting elements. If specified, the locale overrides the locale set at the level of `\DeclareSortingScheme` and also the global setting. See also the discussion of the global sorting locale option `sortlocale` in § 3.1.2.1.

`direction=ascending, descending` default: ascending

The sort direction, which may be either `ascending` or `descending`. The default is `ascending` order.

`final=true, false` default: false

This option marks a `\sort` directive as the final one in the *⟨specification⟩*. If one of the *⟨elements⟩* is available, the remainder of the *⟨specification⟩* will be ignored. The short form `final` is equivalent to `final=true`.

`sortcase=true, false`

Whether or not to sort case-sensitively. The default setting depends on the global `sortcase` option.

`sortupper=true, false`

Whether or not to sort in ‘uppercase before lowercase’ (`true`) or ‘lowercase before uppercase’ order (`false`). The default setting depends on the global `sortupper` option.

`\field[⟨key=value, ...⟩]{⟨field⟩}`

The `\field` element adds a *⟨field⟩* to the sorting specification. If the *⟨field⟩* is undefined, the element is skipped. The `\field` command supports the following optional arguments:

`padside=left, right` default: left

Pads a field on the `left` or `right` side using `padchar` so that its width is `padwidth`. If no padding option is set, no padding is done at all. If any padding option is specified, then padding is performed and the missing options are assigned built-in default values. If padding and substring matching are both specified, the substring match is performed first. Padding is particularly useful with numeric fields. For example, the command

```
\field[padside=left,padwidth=2,padchar=0]{volume}
```

will pad the `volume` field with leading zeros to a width of two characters. This way, volumes are sorted by numeric value (01/02/11/12) rather than in alphabetic order (1/11/12/2).

`padwidth=<integer>` default: 4

The target width in characters.

`padchar=<character>` default: 0

The character to be used when padding the field.

`strside=left, right` default: left

Performs a substring match on the `left` or `right` side of the field. The number of characters to match is specified by the corresponding `strwidth` option. If no substring option is set, no substring matching is performed at all. If any substring option is specified, then substring matching is performed and the missing options are assigned built-in default values. If padding and substring matching are both specified, the substring match is performed first.

`strwidth=<integer>` default: 4

The number of characters to match.

`\literal{<string>}`

The `\literal` element adds a literal `<string>` to the sorting specification. This is useful as a fallback if some fields are not available.

`\citeorder` The `\citeorder` element has a special meaning. It requests a sort based on the lexical order of the actual citations. For entries cited within the same citation command like:

```
\cite{one,two}
```

there is a distinction between the lexical order and the semantic order. Here “one” and “two” have the same semantic order but a unique lexical order. The semantic order only matters if you specify further sorting to disambiguate entries with the same semantic order. For example, this is the definition of the `none` sorting scheme:

```
\DeclareSortingScheme{none}{  
  \sort{\citeorder}  
}
```

This sorts the bibliography purely lexically by the order of the keys in the citation commands. In the example above, it sorts “one” before “two”. However, suppose that you consider “one” and “two” to have the same order (semantic order) since they are cited at the same time and want to further sort these by year. Suppose “two” has an earlier year than “one”:

```
\DeclareSortingScheme{noneyear}{
  \sort{\citeorder}
  \sort{year}
}
```

This sorts “two” before “one”, even though lexically, “one” would sort before “two”. This is possible because the semantic order can be disambiguated by the further sorting on year. With the standard `none` sorting scheme, the lexical order and semantic order are identical because there is nothing further to disambiguate them. This means that you can use `\citeorder` just like any other sorting specification element, choosing how to further sort entries cited at the same time (in the same citation command).

Here are some examples of sorting schemes. In the first example, we define a simple name/title/year scheme. The name element may be either the `author`, the `editor`, or the `translator`. Given this specification, the sorting routine will use the first element which is available and continue with the `title`. Note that the options `use<name>` options are considered automatically in the sorting process:

```
\DeclareSortingScheme{sample}{
  \sort{
    \field{author}
    \field{editor}
    \field{translator}
  }
  \sort{
    \field{title}
  }
  \sort{
    \field{year}
  }
}
```

In the next example, we define the same scheme in a more elaborate way, considering special fields such as `presort`, `sortkey`, `sortname`, etc. Since the `sortkey` field specifies the master sort key, it needs to override all other elements except for `presort`. This is indicated by the `final` option. If the `sortkey` field is available, processing will stop at this point. If not, the sorting routine continues with the next `\sort` directive. This setup corresponds to the default definition of the `nty` scheme:

```
\DeclareSortingScheme{nty}{
  \sort{
    \field{presort}
  }
  \sort[final]{
```

```

    \field{sortkey}
  }
  \sort{
    \field{sortname}
    \field{author}
    \field{editor}
    \field{translator}
    \field{sorttitle}
    \field{title}
  }
  \sort{
    \field{sorttitle}
    \field{title}
  }
  \sort{
    \field{sortyear}
    \field{year}
  }
}

```

Finally, here is an example of a sorting scheme which overrides the global sorting locale and additionally overrides again when sorting by the `origtitle` field. Note the use in the scheme-level override of a babel/polyglossia language name instead of a real locale identifier. Biber will map this to a suitable, real locale identifier (in this case, `sv_SE`):

```

\DeclareSortingScheme[locale=swedish]{custom}{
  \sort{
    \field{sortname}
    \field{author}
    \field{editor}
    \field{translator}
    \field{sorttitle}
    \field{title}
  }
  \sort[locale=de_DE_phonebook]{
    \field{origtitle}
  }
}

```

`\DeclareSortExclusion{⟨entrytype, ...⟩}{⟨field, ...⟩}`

Biber only

Specifies fields to be excluded from sorting on a per-type basis. The `⟨entrytype⟩` argument and the `⟨field⟩` argument may be a comma-separated list of values. A blank `⟨field⟩`

argument will clear all exclusions for this $\langle entrytype \rangle$. This command may only be used in the preamble.

`\DeclarePresort[$\langle entrytype, ... \rangle$]{ $\langle string \rangle$ }`

Biber only

Specifies a string to be used to automatically populate the `presort` field of entries without a `presort` field. The `presort` may be defined globally or on a per-type basis. If the optional $\langle entrytype \rangle$ argument is given, the $\langle string \rangle$ applies to the respective entry type. If not, it serves as the global default value. Specifying an $\langle entrytype \rangle$ in conjunction with a blank $\langle string \rangle$ will clear the type-specific setting. The $\langle entrytype \rangle$ argument may be a comma-separated list of values. This command may only be used in the preamble.

4.5.6 参考文献表过滤器 Bibliography List Filters

Biber only

When using customisable bibliography lists (See § 3.6.4), usually one wants to return in the `.bbl` only those entries which have the particular fields which the bibliography list is summarising. For example, when printing a normal list of shorthands, you want the list returned by Biber in the `.bbl` to contain only those entries which have a `shorthand` field. This is accomplished by defining a bibliography list filter using the `\DeclareBiblistFilter` command. This differs from the filters defined using `\defbibfilter` (see § 3.6.10) since the filters defined by `\defbibfilter` run inside Biblatex after the `.bbl` has been generated. In addition, bibliography lists in the `.bbl` do not contain entry data, only the citation keys for the entries and so no filtering by Biblatex using `\defbibfilter` is possible for bibliography lists.

`\DeclareBiblistFilter{ $\langle name \rangle$ }{ $\langle specification \rangle$ }`

Biber only

Defines a bibliography list filter with $\langle name \rangle$. The $\langle specification \rangle$ consists of one or more `\filter` or `\filteror` macros, all of which must be satisfied for the entry to pass the filter:

`\filter{ $\langle type \rangle$ }{ $\langle filter \rangle$ }`

Filter entries according to the $\langle type \rangle$ and $\langle filter \rangle$. $\langle type \rangle$ can be one of:

type/nottype Entry is/is not of `entrytype` $\langle filter \rangle$

subtype/notsubtype Entry is/is not of `subtype` $\langle filter \rangle$

keyword/notkeyword Entry has/does not have `keyword` $\langle filter \rangle$

field/notfield Entry has/does not have a field called $\langle filter \rangle$

`\filteror{ $\langle type \rangle$ }{ $\langle filters \rangle$ }`

A wrapper around one or more `\filter` commands specifying that they form a disjunctive set, i.e. any one of the $\langle filters \rangle$ must be satisfied.

Fields in the datamodel which are marked as ‘Label fields’ (see § 4.5.3) automatically have a filter defined for them with the same name and which filters out any entries which do not contain the field. For example, Biblatex automatically generates a filter for the shorthand field:

```
\DeclareBibliographyFilter{shorthand}{
  \filter[type=field,filter=shorthand]
}
```

4.5.7 控制姓名首字母生成 Controlling Name Initials Generation

Generating initials for name parts from a given name involves some difficulties when you have names with prefixes, diacritics, hyphens etc. Often, you want to ignore things like prefixes when generating initials so that the initials for “al-Hasan” is just “H” instead of “a-H”. This is tricky when you also have names like “Ho-Pun” where you want the initials to be “H-P”, for example.

```
\DeclareNoinit{<specification>}
```

Biber only

Defines regular expressions to strip from names before generating initials. The *<specification>* is an undelimited list of `\noinit` directives which specify the regular expressions to remove from the name. Spaces, tabs and line endings may be used freely to visually arrange the *<specification>*. Blank lines are not permissible. This command may only be used in the preamble.

```
\noinit{<regexp>}
```

Any number of `\noinit` commands can be given each of which specifies to remove the *<regexp>* from the copy of the name which the initials generation system sees. Since regular expressions usually contain special characters, it is best to enclose them in the provided `\regexp` macro as shown—this will pass the expression through to Biber correctly.

If there is no `\DeclareNoinit` specification, Biber will default to:

```
\DeclareNoinit{
  % strip lowercase prefixes like 'al-' when generating initials from names
  \noinit{\regexp{\b\p{Ll}{2}\p{Pd}}}{
  % strip some common diacritics when generating initials from names
  \noinit{\regexp{[\x{2bf}\x{2018}]}}
}
```

This Biber default strips a couple of diacritics and also strips lowercase prefixes from names before generating initials.

4.5.8 排序微调 Fine Tuning Sorting

对排序微调是很有用的，它可以忽略一些特殊域的某些部分。

`\DeclareNosort{⟨specification⟩}`

Biber only

Defines regular expressions to strip from particular fields or types of fields when sorting. The *⟨specification⟩* is an undelimited list of `\nosort` directives which specify the regular expressions to remove from particular fields or type of field. Spaces, tabs and line endings may be used freely to visually arrange the *⟨specification⟩*. Blank lines are not permissible. This command may only be used in the preamble.

`\nosort{⟨field or field type⟩}{⟨regex⟩}`

Any number of `\nosort` commands can be given each of which specifies to remove the *⟨regex⟩* from the *⟨field⟩* or *⟨field type⟩*. A *⟨field type⟩* is simply a convenience grouping of semantically similar fields from which you might want to remove a regex. Table 9 shows the available field types. Since regular expressions usually contain special characters, it is best to enclose them in the provided `\regex` macro as shown—this will pass the expression through to Biber correctly.

If there is no `\DeclareNosort` specification, Biber will default to:

```
\DeclareNosort{
  % strip prefixes like 'al-' when sorting names
  \nosort{type_names}{\regex{\A\p{L}{2}\p{Pd}}}}
  % strip some diacritics when sorting names
  \nosort{type_names}{\regex{[\x{2bf}\x{2018}]}}
}
```

This Biber default strips a couple of diacritics and also strips prefixes from names when sorting. Suppose you wanted to ignore “The” at the beginning of a `title` field when sorting:

```
\DeclareNosort{
  \nosort{title}{\regex{\AThe\s+}}
}
```

Or if you wanted to ignore “The” at the beginning of any title field:

```
\DeclareNosort{
  \nosort{type_title}{\regex{\AThe\s+}}
}
```

Field Type	Fields
type_name	author
	afterword
	annotator
	bookauthor
	commentator
	editor
	editora
	editorb
	editorc
	foreword
	holder
	introduction
	namea
	nameb
	namec
	shortauthor
	shorteditor
	translator
	booktitle
type_title	eventtitle
	issuetitle
	journaltitle
	maintitle
	origtitle
	title

Table 9: Field types for `\nosort`

4.5.9 特殊域 Special Fields

Some of the automatically generated fields from § 4.2.4.2 may be customized. Note that this requires Biber.

`\DeclareLabelname[⟨entrytype, ...⟩]{⟨specification⟩}`

Biber only

Defines the fields to consider when generating the `labelname` field (see § 4.2.4.2). The *⟨specification⟩* is an ordered list of `\field` commands. The fields are checked in the order listed and the first field which is available will be used as `labelname`. This is the default definition:

```
\DeclareLabelname{%
  \field{shortauthor}
  \field{author}
  \field{shorteditor}
  \field{editor}
  \field{translator}
}
```

The `labelname` field may be customized globally or on a per-type basis. If the optional *⟨entrytype⟩* argument is given, the specification applies to the respective entry type. If

not, it is applied globally. The $\langle entrytype \rangle$ argument may be a comma-separated list of values. This command may only be used in the preamble.

`\DeclareLabeldate[$\langle entrytype, ... \rangle$]{ $\langle specification \rangle$ }`

Biber only

Defines the date components to consider when generating `labelyear`, `labelmonth` and `labelday` fields (see § 4.2.4.2). The $\langle specification \rangle$ is an ordered list of `\field` or `\literal` commands. The items are checked in the order listed and the first item which is available will be used to populate the `labelyear`, `labelmonth` and `labelday` fields. Note that the `\field` items do not have to be datatype ‘date’ in the data model so that you can create pseudo-year labels by, for example, using a `pubstate` field contents, if available, as the year label by defining `\DeclareLabeldate` suitably. Note also that a `\literal` command will always be used when found and so this should always be the last thing in the list. If the value of a `\literal` command is a valid localization string, then this will be resolved in the current language, otherwise the value is used as a literal string as-is. This is the default definition:

```
\DeclareLabeldate{%
  \field{date}
  \field{eventdate}
  \field{origdate}
  \field{urldate}
  \literal{nodate}
}
```

Note that the date field is split by the backend into `year`, `month` which are also valid fields in the default data model. In order to support legacy data which directly sets `year` and/or `month`, the specification ‘date’ in `\DeclareLabeldate` will also match `year` and `month` fields, if present. The `labelyear`, `labelmonth` and `labelday` fields may be customized globally or on a per-type basis. If the optional $\langle entrytype \rangle$ argument is given, the specification applies to the respective entry type. If not, it is applied globally. The $\langle entrytype \rangle$ argument may be a comma-separated list of values. This command may only be used in the preamble. See also § 4.2.4.3.

`\DeclareLabeltitle[$\langle entrytype, ... \rangle$]{ $\langle specification \rangle$ }`

Biber only

Defines the fields to consider when generating the `labeltitle` field (see § 4.2.4.2). The $\langle specification \rangle$ is an ordered list of `\field` commands. The fields are checked in the order listed and the first field which is available will be used as `labeltitle`. This is the default definition:

```
\DeclareLabeltitle{%
  \field{shorttitle}
  \field{title}
}
```

The `labeltitle` field may be customized globally or on a per-type basis. If the optional `<entrytype>` argument is given, the specification applies to the respective entry type. If not, it is applied globally. The `<entrytype>` argument may be a comma-separated list of values. This command may only be used in the preamble.

4.5.10 数据继承 Data Inheritance (`crossref`)

Biber features a highly customizable cross-referencing mechanism with flexible data inheritance rules. This sections deals with the configuration interface. See appendix B for the default configuration. Note that customized data inheritance requires Biber. It will not work with any other backend. A note on terminology: the *child* or *target* is the entry with the `crossref` field, the *parent* or *source* is the entry the `crossref` field points to. The child inherits data from the parent.

`\DefaultInheritance` [`<exceptions>`] { `<options>` }

Biber only

Configures the default inheritance behavior. This command may only be used in the preamble. The default behavior may be customized by setting the following `<options>`:

`all=true, false` default: true

Whether or not to inherit all fields from the parent by default. `all=true` means that the child entry inherits all fields from the parent, unless a more specific inheritance rule has been set up with `\DeclareDataInheritance`. If an inheritance rule is defined for a field, data inheritance is controlled by that rule. `all=false` means that no data is inherited from the parent by default. Each field to be inherited requires an explicit inheritance rule set up with `\DeclareDataInheritance`. The package default is `all=true`.

`override=true, false` default: false

Whether or not to overwrite target fields with source fields if both are defined. This applies both to automatic inheritance and to explicit inheritance rules. The package default is `override=false`, i.e., existing fields of the child entry are not overwritten.

The optional `<exceptions>` are an undelimited list of `\except` directives. Spaces, tabs, and line endings may be used freely to visually arrange the `<exceptions>`. Blank lines are not permissible.

`\except` { `<source>` } { `<target>` } { `<options>` }

Sets the `<options>` for a specific `<source>` and `<target>` combination. The `<source>` and `<target>` arguments specify the parent and the child entry type. The asterisk matches all types and is permissible in either argument.

`\DeclareDataInheritance` { `<source, ...>` } { `<target, ...>` } { `<rules>` }

Biber only

Declares inheritance rules. The `<source>` and `<target>` arguments specify the parent and the child entry type. Either argument may be a single entry type, a comma-separated

list of types, or an asterisk. The asterisk matches all entry types. The *rules* are an undelimited list of `\inherit` and/or `\noinherit` directives. Spaces, tabs, and line endings may be used freely to visually arrange the *rules*. Blank lines are not permissible. This command may only be used in the preamble.

```
\inherit[option]{source}{target}
```

Defines an inheritance rule by mapping a *source* field to a *target* field. The *option* is the override option explained above. When set locally, it takes precedence over any global options set with `\DefaultInheritance`.

```
\noinherit{source}
```

Unconditionally prevents inheritance of the *source* field.

```
\ResetDataInheritance
```

Biber only

Clears all inheritance rules defined with `\DeclareDataInheritance`. This command may only be used in the preamble.

Here are some practical examples:

```
\DefaultInheritance{all=true,override=false}
```

This example shows how to configure the default inheritance behavior. The above settings are the package defaults.

```
\DefaultInheritance[
  \except{*}{online}{all=false}
]{all=true,override=false}
```

This example is similar to the one above but adds one exception: entries of type `@online` will, by default, not inherit any data from any parent.

```
\DeclareDataInheritance{collection}{incollection}{
  \inherit{title}{booktitle}
  \inherit{subtitle}{booksubtitle}
  \inherit{titleaddon}{booktitleaddon}
}
```

So far we have looked at setting up the defaults. For example, `all=true` means that the `publisher` field of a source entry is copied to the `publisher` field of the target entry. In some cases, however, asymmetric mappings are required. They are defined with `\DeclareDataInheritance`. The above example sets up three typical rules for `@incollection` entries referencing a `@collection`. We map the `title` and related fields of the source to the corresponding `booktitle` fields of the target.

```
\DeclareDataInheritance{mvbook,book}{inbook,bookinbook}{
  \inherit{author}{author}
  \inherit{author}{bookauthor}
}
```

This rule is an example of one-to-many mapping: it maps the `author` field of the source to both the `author` and the `bookauthor` fields of the target in order to allow for compact `inbook/bookinbook` entries. The source may be either a `@mvbook` or a `@book` entry, the target either an `@inbook` or a `@bookinbook` entry.

```
\DeclareDataInheritance{*}{inbook,incollection}{
  \noinherit{introduction}
}
```

This rule prevents inheritance of the `introduction` field. It applies to all targets of type `@inbook` or `@incollection`, regardless of the source entry type.

```
\DeclareDataInheritance{*}{*}{
  \noinherit{abstract}
}
```

This rule, which applies to all entries, regardless of the source and target entry types, prevents inheritance of the `abstract` field.

```
\DefaultInheritance{all=true,override=false}
\ResetDataInheritance
```

This example demonstrates how to emulate traditional BibTeX's cross-referencing mechanism. It enables inheritance by default, disables overwriting, and clears all other inheritance rules and mappings.

4.6 辅助命令 Auxiliary Commands

本节工具用来分析和保存参考文献数据而不是格式化或者打印它们。

4.6.1 数据命令 Data Commands

本节的命令允许以 low-level 方式访问未格式化的参考文献数据。这些命令不是用来输出，而是用来将数据保存到临时宏中，可以用于下一步的比较。

`\thefield{<field>}` 展开为未格式化的 `<field>`。如果 `<field>` 未定义那么展开为一个空字符串。

`\strfield{<field>}`

Similar to `\thefield`, except that the field is automatically sanitized such that its value may safely be used in the formation of a control sequence name.

`\csfield{⟨field⟩}`

Similar to `\thefield`, but prevents expansion.

`\usefield{⟨command⟩}{⟨field⟩}`

Executes `⟨command⟩` using the unformatted `⟨field⟩` as its argument.

`\thelist{⟨literal list⟩}`

Expands to the unformatted `⟨literal list⟩`. If the list is undefined, this command expands to an empty string. Note that this command will dump the `⟨literal list⟩` in the internal format used by this package. This format is not suitable for printing.

`\thename{⟨name list⟩}`

Expands to the unformatted `⟨name list⟩`. If the list is undefined, this command expands to an empty string. Note that this command will dump the `⟨name list⟩` in the internal format used by this package. This format is not suitable for printing.

`\savefield{⟨field⟩}{⟨macro⟩}`

`\savefield*{⟨field⟩}{⟨macro⟩}`

Copies an unformatted `⟨field⟩` to a `⟨macro⟩`. The regular variant of this command defines the `⟨macro⟩` globally, the starred one works locally.

`\savelist{⟨literal list⟩}{⟨macro⟩}`

`\savelist*{⟨literal list⟩}{⟨macro⟩}`

Copies an unformatted `⟨literal list⟩` to a `⟨macro⟩`. The regular variant of this command defines the `⟨macro⟩` globally, the starred one works locally.

`\savename{⟨name list⟩}{⟨macro⟩}`

`\savename*{⟨name list⟩}{⟨macro⟩}`

Copies an unformatted `⟨name list⟩` to a `⟨macro⟩`. The regular variant of this command defines the `⟨macro⟩` globally, the starred one works locally.

`\savefieldcs{⟨field⟩}{⟨csname⟩}`

`\savefieldcs*{⟨field⟩}{⟨csname⟩}`

Similar to `\savefield`, but takes the control sequence name `⟨csname⟩` (without a leading backslash) as an argument, rather than a macro name.

`\savelistcs{⟨literal list⟩}{⟨csname⟩}`

`\savelistcs*{⟨literal list⟩}{⟨csname⟩}`

Similar to `\savelist`, but takes the control sequence name `⟨csname⟩` (without a leading backslash) as an argument, rather than a macro name.

```
\savenamecs{⟨name list⟩}{⟨csname⟩}  
\savenamecs*{⟨name list⟩}{⟨csname⟩}
```

Similar to `\savename`, but takes the control sequence name `⟨csname⟩` (without a leading backslash) as an argument, rather than a macro name.

```
\restorefield{⟨field⟩}{⟨macro⟩}
```

Restores a `⟨field⟩` from a `⟨macro⟩` defined with `\savefield` before. The field is restored within a local scope.

```
\restorelist{⟨literal list⟩}{⟨macro⟩}
```

Restores a `⟨literal list⟩` from a `⟨macro⟩` defined with `\savelist` before. The list is restored within a local scope.

```
\restorename{⟨name list⟩}{⟨macro⟩}
```

Restores a `⟨name list⟩` from a `⟨macro⟩` defined with `\savename` before. The list is restored within a local scope.

```
\clearfield{⟨field⟩}
```

Clears the `⟨field⟩` within a local scope. A field cleared this way is treated as undefined by subsequent data commands.

```
\clearlist{⟨literal list⟩}
```

Clears the `⟨literal list⟩` within a local scope. A list cleared this way is treated as undefined by subsequent data commands.

```
\clearname{⟨name list⟩}
```

Clears the `⟨name list⟩` within a local scope. A list cleared this way is treated as undefined by subsequent data commands.

4.6.2 独立判断命令 Stand-alone Tests

The commands in this section are various kinds of stand-alone tests for use in bibliography and citation styles.

```
\iffieldundef{⟨field⟩}{⟨true⟩}{⟨false⟩}
```

Expands to `⟨true⟩` if the `⟨field⟩` is undefined, and to `⟨false⟩` otherwise.

```
\iflistundef{⟨literal list⟩}{⟨true⟩}{⟨false⟩}
```

Expands to `⟨true⟩` if the `⟨literal list⟩` is undefined, and to `⟨false⟩` otherwise.

```
\ifnameundef{⟨name list⟩}{⟨true⟩}{⟨false⟩}
```

Expands to `⟨true⟩` if the `⟨name list⟩` is undefined, and to `⟨false⟩` otherwise.

`\iffieldequal`{ $\langle field\ 1 \rangle$ }{ $\langle field\ 2 \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Expands to $\langle true \rangle$ if the values of $\langle field\ 1 \rangle$ and $\langle field\ 2 \rangle$ are equal, and to $\langle false \rangle$ otherwise.

`\iflistsequal`{ $\langle literal\ list\ 1 \rangle$ }{ $\langle literal\ list\ 2 \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Expands to $\langle true \rangle$ if the values of $\langle literal\ list\ 1 \rangle$ and $\langle literal\ list\ 2 \rangle$ are equal, and to $\langle false \rangle$ otherwise.

`\ifnameequal`{ $\langle name\ list\ 1 \rangle$ }{ $\langle name\ list\ 2 \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Expands to $\langle true \rangle$ if the values of $\langle name\ list\ 1 \rangle$ and $\langle name\ list\ 2 \rangle$ are equal, and to $\langle false \rangle$ otherwise.

`\iffieldequals`{ $\langle field \rangle$ }{ $\langle macro \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Expands to $\langle true \rangle$ if the value of the $\langle field \rangle$ is equal to the definition of $\langle macro \rangle$, and to $\langle false \rangle$ otherwise.

`\iflistequals`{ $\langle literal\ list \rangle$ }{ $\langle macro \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Expands to $\langle true \rangle$ if the value of the $\langle literal\ list \rangle$ is equal to the definition of $\langle macro \rangle$, and to $\langle false \rangle$ otherwise.

`\ifnameequals`{ $\langle name\ list \rangle$ }{ $\langle macro \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Expands to $\langle true \rangle$ if the value of the $\langle name\ list \rangle$ is equal to the definition of $\langle macro \rangle$, and to $\langle false \rangle$ otherwise.

`\iffieldequalcs`{ $\langle field \rangle$ }{ $\langle csname \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Similar to `\iffieldequals` but takes the control sequence name $\langle csname \rangle$ (without a leading backslash) as an argument, rather than a macro name.

`\iflistequalcs`{ $\langle literal\ list \rangle$ }{ $\langle csname \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Similar to `\iflistequals` but takes the control sequence name $\langle csname \rangle$ (without a leading backslash) as an argument, rather than a macro name.

`\ifnameequalcs`{ $\langle name\ list \rangle$ }{ $\langle csname \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Similar to `\ifnameequals` but takes the control sequence name $\langle csname \rangle$ (without a leading backslash) as an argument, rather than a macro name.

`\iffieldequalstr`{ $\langle field \rangle$ }{ $\langle string \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }

Executes $\langle true \rangle$ if the value of the $\langle field \rangle$ is equal to $\langle string \rangle$, and $\langle false \rangle$ otherwise. This command is robust.

`\iffieldxref{⟨field⟩}{⟨true⟩}{⟨false⟩}`

If the `crossref/xref` field of an entry is defined, this command checks if the `⟨field⟩` is related to the cross-referenced parent entry. It executes `⟨true⟩` if the `⟨field⟩` of the child entry is equal to the corresponding `⟨field⟩` of the parent entry, and `⟨false⟩` otherwise. If the `crossref/xref` field is undefined, it always executes `⟨false⟩`. This command is robust. See the description of the `crossref` and `xref` fields in § 2.2.3 as well as § 2.4.1 for further information concerning cross-referencing.

`\iflistxref{⟨literal list⟩}{⟨true⟩}{⟨false⟩}`

Similar to `\iffieldxref` but checks if a `⟨literal list⟩` is related to the cross-referenced parent entry. See the description of the `crossref` and `xref` fields in § 2.2.3 as well as § 2.4.1 for further information concerning cross-referencing.

`\ifnamexref{⟨name list⟩}{⟨true⟩}{⟨false⟩}`

Similar to `\iffieldxref` but checks if a `⟨name list⟩` is related to the cross-referenced parent entry. See the description of the `crossref` and `xref` fields in § 2.2.3 as well as § 2.4.1 for further information concerning cross-referencing.

`\ifcurrentfield{⟨field⟩}{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if the current field is `⟨field⟩`, and `⟨false⟩` otherwise. This command is robust. It is intended for use in field formatting directives and always executes `⟨false⟩` when used in any other context.

`\ifcurrentlist{⟨literal list⟩}{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if the current list is `⟨literal list⟩`, and `⟨false⟩` otherwise. This command is robust. It is intended for use in list formatting directives and always executes `⟨false⟩` when used in any other context.

`\ifcurrentname{⟨name list⟩}{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if the current list is `⟨name list⟩`, and `⟨false⟩` otherwise. This command is robust. It is intended for use in list formatting directives and always executes `⟨false⟩` when used in any other context.

`\ifuseprefix{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` if the `useprefix` option is enabled (either globally or for the current entry), and `⟨false⟩` otherwise. See § 3.1.3 for details on this option.

`\ifuseauthor{⟨true⟩}{⟨false⟩}`

This is just a particular case of the `\ifuse<name>` macro below but is mentioned here as `author` is part of the default data model. Expands to `⟨true⟩` if the `useauthor` option is enabled (either globally or for the current entry), and `⟨false⟩` otherwise. See § 3.1.3 for details on this option.

```
\ifuseeditor{⟨true⟩}{⟨false⟩}
```

This is just a particular case of the `\ifuse<name>` macro below but is mentioned here as `editor` is part of the default data model. Expands to `⟨true⟩` if the `useeditor` option is enabled (either globally or for the current entry), and `⟨false⟩` otherwise. See § 3.1.3 for details on this option.

```
\ifusetranslator{⟨true⟩}{⟨false⟩}
```

This is just a particular case of the `\ifuse<name>` macro below but is mentioned here as `translator` is part of the default data model. Expands to `⟨true⟩` if the `usetranslator` option is enabled (either globally or for the current entry), and `⟨false⟩` otherwise. See § 3.1.3 for details on this option.

```
\ifuse<name>{⟨true⟩}{⟨false⟩}
```

Expands to `⟨true⟩` if the `use<name>` option is enabled (either globally or for the current entry), and `⟨false⟩` otherwise. See § 3.1.3 for details on this option.

```
\ifsingletitle{⟨true⟩}{⟨false⟩}
```

Expands to `⟨true⟩` if there is only one work by the `labelname` name in the bibliography, and to `⟨false⟩` otherwise. If there is no `labelname` name at all for the entry, then this expands to `⟨true⟩` if there is only one work with the `labeltitle` title in the bibliography and `⟨false⟩` otherwise. If neither `labelname` nor `labeltitle` are set for an entry, this will always expand to `⟨false⟩`. Note that this feature needs to be enabled explicitly with the package option `singletitle`.

```
\ifandothers{⟨list⟩}{⟨true⟩}{⟨false⟩}
```

Expands to `⟨true⟩` if the `⟨list⟩` is defined and has been truncated in the `bib` file with the keyword ‘and others’, and to `⟨false⟩` otherwise. The `⟨list⟩` may be a literal list or a name list.

```
\ifmorenames{⟨true⟩}{⟨false⟩}
```

Expands to `⟨true⟩` if the current name list has been or will be truncated, and to `⟨false⟩` otherwise. This command is intended for use in formatting directives for name lists. It will always expand to `⟨false⟩` when used elsewhere. This command performs the equivalent of an `\ifandothers` test for the current list. If this test is negative, it also checks if the `listtotal` counter is larger than `liststop`. This command may be used in a formatting directive to decide if a note such as “and others” or “et al.” is to be printed at the end of the list. Note that you still need to check whether you are in the middle or at the end of the list, i.e., whether `listcount` is smaller than or equal to `liststop`, see § 4.4.1 for details.

`\ifmoreitems{⟨true⟩}{⟨false⟩}`

This command is similar to `\ifmorenames` but checks the current literal list. It is intended for use in formatting directives for literal lists. It will always expand to `⟨false⟩` when used elsewhere.

`\iffirstinits{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` or `⟨false⟩`, depending on the state of the `firstinits` package option (see § 3.1.2.3). This command is intended for use in formatting directives for name lists.

`\ifterseinits{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` or `⟨false⟩`, depending on the state of the `terseinits` package option (see § 3.1.2.3). This command is intended for use in formatting directives for name lists.

`\ifentrytype{⟨type⟩}{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if the entry type of the entry currently being processed is `⟨type⟩`, and `⟨false⟩` otherwise.

`\ifkeyword{⟨keyword⟩}{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if the `⟨keyword⟩` is found in the `keywords` field of the entry currently being processed, and `⟨false⟩` otherwise.

`\ifentrykeyword{⟨entrykey⟩}{⟨keyword⟩}{⟨true⟩}{⟨false⟩}`

A variant of `\ifkeyword` which takes an entry key as its first argument. This is useful for testing an entry other than the one currently processed.

`\ifcategory{⟨category⟩}{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if the entry currently being processed has been assigned to a `⟨category⟩` with `\addtocategory`, and `⟨false⟩` otherwise.

`\ifentrycategory{⟨entrykey⟩}{⟨category⟩}{⟨true⟩}{⟨false⟩}`

A variant of `\ifcategory` which takes an entry key as its first argument. This is useful for testing an entry other than the one currently processed.

`\ifciteseen{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if the entry currently being processed has been cited before, and `⟨false⟩` otherwise. This command is robust and intended for use in citation styles. If there are any `refsection` environments in the document, the citation tracking is local to these environments. Note that the citation tracker needs to be enabled explicitly with the package option `citetracker`. The behavior of this test depends on the

mode the citation tracker is operating in, see § 3.1.2.3 for details. If the citation tracker is disabled, the test always yields *⟨false⟩*. Also see the `\citetrackertrue` and `\citetrackerfalse` switches in § 4.6.4.

```
\ifentryseen{⟨entrykey⟩}{⟨true⟩}{⟨false⟩}
```

A variant of `\ifciteseen` which takes an entry key as its first argument. Since the *⟨entrykey⟩* is expanded prior to performing the test, it is possible to test for entry keys in a field such as `xref`:

```
\ifentryseen{\thefield{xref}}{true}{false}
```

Apart from the additional argument, `\ifentryseen` behaves like `\ifciteseen`.

```
\ifentryinbib{⟨entrykey⟩}{⟨true⟩}{⟨false⟩}
```

Executes *⟨true⟩* if the entry *⟨entrykey⟩* appears in the current bibliography, and *⟨false⟩* otherwise. This command is intended for use in bibliography styles.

```
\iffirstcitekey{⟨true⟩}{⟨false⟩}
```

Executes *⟨true⟩* if the entry currently being processed is the first one in the citation list, and *⟨false⟩* otherwise. This command relies on the `citecount`, `citetotal`, `multicitecount` and `multicitetotal` counters (§ 4.10.5) and thus is intended for use only in the *⟨loopcode⟩* of a citation command defined with `\DeclareCiteCommand`.

```
\iflastcitekey{⟨true⟩}{⟨false⟩}
```

Similar `\iffirstcitekey`, but executes *⟨true⟩* if the entry currently being processed is the last one in the citation list, and *⟨false⟩* otherwise.

```
\ifciteibid{⟨true⟩}{⟨false⟩}
```

Expands to *⟨true⟩* if the entry currently being processed is the same as the last one, and to *⟨false⟩* otherwise. This command is intended for use in citation styles. If there are any `refsection` environments in the document, the tracking is local to these environments. Note that the ‘ibidem’ tracker needs to be enabled explicitly with the package option `ibidtracker`. The behavior of this test depends on the mode the tracker is operating in, see § 3.1.2.3 for details. If the tracker is disabled, the test always yields *⟨false⟩*. Also see the `\citetrackertrue` and `\citetrackerfalse` switches in § 4.6.4.

```
\ifciteidem{⟨true⟩}{⟨false⟩}
```

Expands to *⟨true⟩* if the primary name (i. e., the author or editor) in the entry currently being processed is the same as the last one, and to *⟨false⟩* otherwise. This command is intended for use in citation styles. If there are any `refsection` environments in

the document, the tracking is local to these environments. Note that the ‘idem’ tracker needs to be enabled explicitly with the package option `idemtracker`. The behavior of this test depends on the mode the tracker is operating in, see § 3.1.2.3 for details. If the tracker is disabled, the test always yields `\false`. Also see `\citetrackertrue` and `\citetrackerfalse` in § 4.6.4.

```
\ifopcit{\true}{\false}
```

This command is similar to `\ifciteibid` except that it expands to `\true` if the entry currently being processed is the same as the last one *by this author or editor*. Note that the ‘opcit’ tracker needs to be enabled explicitly with the package option `opcittracker`. The behavior of this test depends on the mode the tracker is operating in, see § 3.1.2.3 for details. If the tracker is disabled, the test always yields `\false`. Also see the `\citetrackertrue` and `\citetrackerfalse` switches in § 4.6.4.

```
\ifloccit{\true}{\false}
```

This command is similar to `\ifopcit` except that it also compares the `\postnote` arguments and expands to `\true` only if they match and are numerical (in the sense of `\ifnumerals` from § 4.6.2), i.e., `\ifloccit` will yield `\true` if the citation refers to the same page cited before. Note that the ‘loccit’ tracker needs to be enabled explicitly with the package option `loccittracker`. The behavior of this test depends on the mode the tracker is operating in, see § 3.1.2.3 for details. If the tracker is disabled, the test always yields `\false`. Also see the `\citetrackertrue` and `\citetrackerfalse` switches in § 4.6.4.

```
\iffirstonpage{\true}{\false}
```

The behavior of this command is responsive to the package option `pagetracker`. If the option is set to `page`, it expands to `\true` if the current item is the first one on the page, and to `\false` otherwise. If the option is set to `spread`, it expands to `\true` if the current item is the first one on the double-page spread, and to `\false` otherwise. If the page tracker is disabled, this test always yields `\false`. Depending on the context, the ‘item’ may be a citation or an entry in the bibliography or a bibliography list. Note that this test distinguishes between body text and footnotes. For example, if used in the first footnote on a page, it will expand to `\true` even if there is a citation in the body text prior to the footnote. Also see the `\pagetrackertrue` and `\pagetrackerfalse` switches in § 4.6.4.

```
\ifsamepage{\instance 1}{\instance 2}{\true}{\false}
```

This command expands to `\true` if two instances of a reference are located on the same page or double-page spread, and to `\false` otherwise. An instance of a reference may be a citation or an entry in the bibliography or a bibliography list. These instances are identified by the value of the `instcount` counter, see § 4.10.5. The behavior of this command is responsive to the package option `pagetracker`. If this option is set

to spread, `\ifsamepage` is in fact an ‘if same spread’ test. If the page tracker is disabled, this test always yields *false*. The arguments *instance 1* and *instance 2* are treated as integer expressions in the sense of e-TeX’s `\numexpr`. This implies that it is possible to make calculations within these arguments, for example:

```
\ifsamepage{\value{instcount}}{\value{instcount}-1}{true}{false}
```

Note that `\value` is not prefixed by `\the` and that the subtraction is included in the second argument in the above example. If *instance 1* or *instance 2* is an invalid number (for example, a negative one), the test yields *false*. Also note that this test does not distinguish between body text and footnotes. Also see the `\pagetrackertrue` and `\pagetrackerfalse` switches in § 4.6.4.

```
\ifinteger{<string>}{<true>}{<false>}
```

Executes *true* if the *string* is a positive integer, and *false* otherwise. This command is robust.

```
\ifnumeral{<string>}{<true>}{<false>}
```

Executes *true* if the *string* is an Arabic or Roman numeral, and *false* otherwise. This command is robust. See also `\DeclareNumChars` and `\NumCheckSetup` in § 4.6.4.

```
\ifnumerals{<string>}{<true>}{<false>}
```

Executes *true* if the *string* is a range or a list of Arabic or Roman numerals, and *false* otherwise. This command is robust. In contrast to `\ifnumeral`, it will also execute *true* with arguments like “52–58”, “14/15”, “1, 3, 5”, and so on. See also `\DeclareNumChars`, `\DeclareRangeChars`, `\DeclareRangeCommands`, and `\NumCheckSetup` in § 4.6.4.

```
\ifpages{<string>}{<true>}{<false>}
```

Similar to `\ifnumerals`, but also considers `\DeclarePageCommands` from § 4.6.4.

```
\iffieldint{<field>}{<true>}{<false>}
```

Similar to `\ifinteger`, but uses the value of a *field* rather than a literal string in the test. If the *field* is undefined, it executes *false*.

```
\iffieldnum{<field>}{<true>}{<false>}
```

Similar to `\ifnumeral`, but uses the value of a *field* rather than a literal string in the test. If the *field* is undefined, it executes *false*.

```
\iffieldnums{<field>}{<true>}{<false>}
```

Similar to `\ifnumerals`, but uses the value of a *field* rather than a literal string in the test. If the *field* is undefined, it executes *false*.

`\iffieldpages{⟨field⟩}{⟨true⟩}{⟨false⟩}`

Similar to `\ifpages`, but uses the value of a `⟨field⟩` rather than a literal string in the test. If the `⟨field⟩` is undefined, it executes `⟨false⟩`.

`\ifbibstring{⟨string⟩}{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` if the `⟨string⟩` is a known localization key, and to `⟨false⟩` otherwise. The localization keys defined by default are listed in § 4.9.2. New ones may be defined with `\NewBibliographyString`.

`\ifbibxstring{⟨string⟩}{⟨true⟩}{⟨false⟩}`

Similar to `\ifbibstring`, but the `⟨string⟩` is expanded.

`\iffieldbibstring{⟨field⟩}{⟨true⟩}{⟨false⟩}`

Similar to `\ifbibstring`, but uses the value of a `⟨field⟩` rather than a literal string in the test. If the `⟨field⟩` is undefined, it expands to `⟨false⟩`.

`\ifdriver{⟨entrytype⟩}{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` if a driver for the `⟨entrytype⟩` is available, and to `⟨false⟩` otherwise.

`\ifcapital{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if Bibl_{at}ex’s punctuation tracker would capitalize a localization string at the current location, and `⟨false⟩` otherwise. This command is robust. It may be useful for conditional capitalization of certain parts of a name in a formatting directive.

`\ifcitation{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` when located in a citation, and to `⟨false⟩` otherwise. Note that this command is responsive to the outermost context in which it is used. For example, if a citation command defined with `\DeclareCiteCommand` executes a driver defined with `\DeclareBibliographyDriver`, any `\ifcitation` tests in the driver code will yield `⟨true⟩`. See § 4.11.6 for a practical example.

`\ifbibliography{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` when located in a bibliography, and to `⟨false⟩` otherwise. Note that this command is responsive to the outermost context in which it is used. For example, if a driver defined with `\DeclareBibliographyDriver` executes a citation command defined with `\DeclareCiteCommand`, any `\ifbibliography` tests in the citation code will yield `⟨true⟩`. See § 4.11.6 for a practical example.

`\ifnatbibmode{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` or `⟨false⟩` depending on the `natbib` option from § 3.1.1.

`\ifciteindex{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` or `⟨false⟩` depending on the indexing option from § 3.1.2.1.

`\ifbibindex{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` or `⟨false⟩` depending on the indexing option from § 3.1.2.1.

`\iffootnote{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` when located in a footnote, and to `⟨false⟩` otherwise. Note that footnotes in `minipage` environments are considered to be part of the body text. This command will only expand to `⟨true⟩` in footnotes at the bottom of the page and in endnotes as provided by the `endnotes` package.

citecounter This counter indicates how many times the entry currently being processed is cited in the current reference section. Note that this feature needs to be enabled explicitly with the package option `citecounter`. If the option is set to `context`, citations in the body text and in footnotes are counted separately. In this case, `citecounter` will hold the value of the context it is used in.

uniquename This counter refers to the `labelname` list. It is set on a per-name basis. Its value is 0 if the last name is unique, 1 if adding the other parts of the name (first name, prefix, suffix) as initials will make it unique, and 2 if the full name is required to disambiguate the name. This information is required by author-year and author-title citation schemes which add additional parts of the name when citing different authors with the same last name. For example, if there is one ‘John Doe’ and one ‘Edward Doe’ in the list of references, this counter will be set to 1. If there is one ‘John Doe’ and one ‘Jane Doe’, the value of the counter will be 2. If the option is set to `init/allinit/mininit`, the counter will be limited to 1. This is useful for citations styles which use initials to disambiguate names but never print the full name in citations. If adding the initials is not sufficient to disambiguate the name, `uniquename` will also be set to 0 for that name. This feature needs to be enabled explicitly with the package option `uniquename`. Note that the `uniquename` counter is local to `\printnames` and that it is only set for the `labelname` list and to the name list `labelname` has been derived from (typically `author` or `editor`). Its value is zero in any other context, i.e., it must be evaluated in the name formatting directives handling name lists. See § 4.11.4 for further details and practical examples.

Biber only

uniquelist This counter refers to the `labelname` list. It is set on a per-field basis. Its value indicates the number of names required to disambiguate the name list if automatic `maxnames/minnames` truncation would lead to ambiguous citations. For example, if there is one work by ‘Doe/Smith/Johnson’ and another one by ‘Doe/Edwards/Williams’, setting `maxnames=1` would lead to ‘Doe et al.’ in both cases. In this case, `uniquelist` would be set to 2 on the `labelname` lists of both entries because at least the first two names are required to disambiguate them. Note that

Biber only

the `uniquelist` counter is local to `\printrnames` and that it is only set for the `labelname` list and to the name list `labelname` has been derived from (typically author or editor). Its value is zero in any other context. If available, the `uniquelist` value will be used automatically by `\printrnames` when processing the name list, i.e., it will automatically override `maxnames/minnames`. This feature needs to be enabled explicitly with the package option `uniquelist`. See § 4.11.4 for further details and practical examples.

`parenlevel` The current nesting level of parentheses and/or brackets. This information is only available if the `parenttracker` from § 3.1.2.3 is enabled.

4.6.3 使用`\ifboolexpr`和`\ifthenelse`的判断 Tests with `\ifboolexpr` and `\ifthenelse`

The tests introduced in § 4.6.2 may also be used with the `\ifboolexpr` command provided by the `etoolbox` package and the `\ifthenelse` command provided by the `ifthen` package. The syntax of the tests is slightly different in this case: the `<true>` and `<false>` arguments are omitted from the test itself and passed to the `\ifboolexpr` or `\ifthenelse` command instead. Note that the use of these commands implies some processing overhead. If you do not need any boolean operators, it is more efficient to use the stand-alone tests from § 4.6.2.

`\ifboolexpr{<expression>}{<true>}{<false>}`

`etoolbox` command which allows for complex tests with boolean operators and grouping:

```
\ifboolexpr{ (
    test {\ifnameundef{editor}}
    and
    not test {\iflistundef{location}}
)
or test {\iffieldundef{year}}
}
{...}
{...}
```

`\ifthenelse{<tests>}{<true>}{<false>}`

`ifthen` command which allows for complex tests with boolean operators and grouping:

```
\ifthenelse{ \ (
    \ifnameundef{editor}
    \and
```

```

        \not \iflistundef{location}
    \)
    \or \iffieldundef{year}
}
{...}
{...}

```

The additional tests provided by Biblatex are only available when `\ifboolexpr` or `\ifthenelse` are used in citation commands and in the bibliography.

4.6.4 综合命令 Miscellaneous Commands

The section introduced miscellaneous commands and little helpers for use in bibliography and citation styles.

```

\newbibmacro{<name>}[<arguments>][<optional>]{<definition>}
\newbibmacro*{<name>}[<arguments>][<optional>]{<definition>}

```

Defines a macro to be executed via `\usebibmacro` later. The syntax of this command is very similar to `\newcommand` except that `<name>` may contain characters such as numbers and punctuation marks and does not start with a backslash. The optional argument `<arguments>` is an integer specifying the number of arguments taken by the macro. If `<optional>` is given, it specifies a default value for the first argument of the macro, which automatically becomes an optional argument. In contrast to `\newcommand`, `\newbibmacro` issues a warning message if the macro is already defined, and automatically falls back to `\renewbibmacro`. As with `\newcommand`, the regular variant of this command uses the `\long` prefix in the definition while the starred one does not. If a macro has been declared to be long, it may take arguments containing `\par` tokens. `\newbibmacro` and `\renewbibmacro` are provided for convenience. Style authors are free to use `\newcommand` or `\def` instead. However, note that most shared definitions found in `biblatex.def` are defined with `\newbibmacro`, hence they must be used and modified accordingly.

```

\renewbibmacro{<name>}[<arguments>][<optional>]{<definition>}
\renewbibmacro*{<name>}[<arguments>][<optional>]{<definition>}

```

Similar to `\newbibmacro` but redefines `<name>`. In contrast to `\renewcommand`, `\renewbibmacro` issues a warning message if the macro is undefined, and automatically falls back to `\newbibmacro`.

```

\providebibmacro{<name>}[<arguments>][<optional>]{<definition>}
\providebibmacro*{<name>}[<arguments>][<optional>]{<definition>}

```

Similar to `\newbibmacro` but only defines `<name>` if it is undefined. This command is similar in concept to `\providecommand`.

```
\usebibmacro{⟨name⟩}  
\usebibmacro*{⟨name⟩}
```

This command executes the macro $\langle name \rangle$, as defined with `\newbibmacro`. If the macro takes any arguments, they are simply appended after $\langle name \rangle$. The regular variant of this command sanitizes $\langle name \rangle$ while the starred variant does not.

```
\savecommand{⟨command⟩}  
\restorecommand{⟨command⟩}
```

These commands save and restore any $\langle command \rangle$, which must be a command name starting with a backslash. Both commands work within a local scope. They are mainly provided for use in localization files.

```
\savebibmacro{⟨name⟩}  
\restorebibmacro{⟨name⟩}
```

These commands save and restore the macro $\langle name \rangle$, where $\langle name \rangle$ is the identifier of a macro defined with `\newbibmacro`. Both commands work within a local scope. They are mainly provided for use in localization files.

```
\savefieldformat[⟨entry type⟩]{⟨format⟩}  
\restorefieldformat[⟨entry type⟩]{⟨format⟩}
```

These commands save and restore the formatting directive $\langle format \rangle$, as defined with `\DeclareFieldFormat`. Both commands work within a local scope. They are mainly provided for use in localization files.

```
\savelistformat[⟨entry type⟩]{⟨format⟩}  
\restorelistformat[⟨entry type⟩]{⟨format⟩}
```

These commands save and restore the formatting directive $\langle format \rangle$, as defined with `\DeclareListFormat`. Both commands work within a local scope. They are mainly provided for use in localization files.

```
\savenameformat[⟨entry type⟩]{⟨format⟩}  
\restorenameformat[⟨entry type⟩]{⟨format⟩}
```

These commands save and restore the formatting directive $\langle format \rangle$, as defined with `\DeclareNameFormat`. Both commands work within a local scope. They are mainly provided for use in localization files.

```
\ifbibmacroundef{⟨name⟩}{⟨true⟩}{⟨false⟩}
```

Expands to $\langle true \rangle$ if the bibliography macro $\langle name \rangle$ is undefined, and to $\langle false \rangle$ otherwise.

```

\iffielddformatundef[⟨entry type⟩]{⟨name⟩}{⟨true⟩}{⟨false⟩}
\iflistformatundef[⟨entry type⟩]{⟨name⟩}{⟨true⟩}{⟨false⟩}
\ifnameformatundef[⟨entry type⟩]{⟨name⟩}{⟨true⟩}{⟨false⟩}

```

Expands to $\langle true \rangle$ if the formatting directive $\langle format \rangle$ is undefined, and to $\langle false \rangle$ otherwise.

```

\usedriver{⟨code⟩}{⟨entrytype⟩}

```

Executes the bibliography driver for an $\langle entrytype \rangle$. Calling this command in the $\langle loopcode \rangle$ of a citation command defined with `\DeclareCiteCommand` is a simple way to print full citations similar to a bibliography entry. Commands such as `\newblock`, which are not applicable in a citation, are disabled automatically. Additional initialization commands may be passed as the $\langle code \rangle$ argument. This argument is executed inside the group in which `\usedriver` runs the respective driver. Note that it is mandatory in terms of the syntax but may be left empty. Also note that this command will automatically switch languages if the `autolang` package option is enabled.

```

\bibhypertarget{⟨name⟩}{⟨text⟩}

```

A wrapper for `hyperref`'s `\hypertarget` command. The $\langle name \rangle$ is the name of the anchor, the $\langle text \rangle$ is arbitrary printable text or code which serves as an anchor. If there are any `refsection` environments in the document, the $\langle name \rangle$ is local to the current environment. If the `hyperref` package option is disabled or the `hyperref` package has not been loaded, this command will simply pass on its $\langle text \rangle$ argument. See also the formatting directive `bibhypertarget` in § 4.10.4.

```

\bibhyperlink{⟨name⟩}{⟨text⟩}

```

A wrapper for `hyperref`'s `\hyperlink` command. The $\langle name \rangle$ is the name of an anchor defined with `\bibhypertarget`, the $\langle text \rangle$ is arbitrary printable text or code to be transformed into a link. If there are any `refsection` environments in the document, the $\langle name \rangle$ is local to the current environment. If the `hyperref` package option is disabled or the `hyperref` package has not been loaded, this command will simply pass on its $\langle text \rangle$ argument. See also the formatting directive `bibhyperlink` in § 4.10.4.

```

\bibhyperref[⟨entrykey⟩]{⟨text⟩}

```

Transforms $\langle text \rangle$ into an internal link pointing to $\langle entrykey \rangle$ in the bibliography. If $\langle entrykey \rangle$ is omitted, this command uses the key of the entry currently being processed. This command is employed to transform citations into clickable links pointing to the corresponding entry in the bibliography. The link target is marked automatically by Biblatex. If there are multiple bibliographies in a document, the target will be the first occurrence of $\langle entrykey \rangle$ in one of the bibliographies. If there are `refsection`

environments, the links are local to the environment. See also the formatting directive `bibhyperref` in § 4.10.4.

`\ifhyperref{⟨true⟩}{⟨false⟩}`

Expands to `⟨true⟩` if the `hyperref` package option is enabled (which implies that the `hyperref` package has been loaded), and to `⟨false⟩` otherwise.

`\docsvfield{⟨field⟩}`

Similar to the `\docsvlist` command from the `etoolbox` package, except that it takes a field name as its argument. The value of this field is parsed as a comma-separated list. If the `⟨field⟩` is undefined, this command expands to an empty string.

`\forcsvfield{⟨handler⟩}{⟨field⟩}`

Similar to the `\forcsvlist` command from the `etoolbox` package, except that it takes a field name as its argument. The value of this field is parsed as a comma-separated list. If the `⟨field⟩` is undefined, this command expands to an empty string.

`\MakeCapital{⟨text⟩}`

Similar to `\MakeUppercase` but only converts the first printable character in `⟨text⟩` to uppercase. Note that the restrictions that apply to `\MakeUppercase` also apply to this command. Namely, all commands in `⟨text⟩` must either be robust or prefixed with `\protect` since the `⟨text⟩` is expanded during capitalization. Apart from Ascii characters and the standard accent commands, this command also handles the active characters of the `inputenc` package as well as the shorthands of the `babel` package. If the `⟨text⟩` starts with a control sequence, nothing is capitalized. This command is robust.

`\MakeSentenceCase{⟨text⟩}`

`\MakeSentenceCase*{⟨text⟩}`

Converts its `⟨text⟩` argument to sentence case, i. e., the first word is capitalized and the remainder of the string is converted to lowercase. This command is robust. The starred variant differs from the regular version in that it considers the language of the entry, as specified in the `langid` field. It only converts the `⟨text⟩` to sentence case if the `langid` field is undefined or if it holds a language declared with `\DeclareCaseLangs` (see below).³⁷ Otherwise, the `⟨text⟩` is not altered in any way. It is recommended to use `\MakeSentenceCase*` rather than the regular variant in formatting directives. Both variants support the traditional BibTeX convention for `bib` files that anything wrapped in a pair of curly braces is not modified when changing the case. For example:

³⁷By default, converting to sentence case is enabled for the following language identifiers: `american`, `british`, `canadian`, `english`, `australian`, `newzealand` as well as the aliases `USenglish` and `UKenglish`. Use `\DeclareCaseLangs` to extend or change this list.


```
\MakeSentenceCase{an Introduction to LaTeX}
\MakeSentenceCase{an Introduction to {LaTeX}}
```

would yield:

```
An introduction to latex
An introduction to LaTeX
```

In bib files designed with traditional BibTeX in mind, it has been fairly common to only wrap single letters in braces to prevent case-changing:

```
title = {An Introduction to {L}a{T}e{X}}
```

The problem with this convention is that the braces will suppress the kerning on both sides of the enclosed letter. It is preferable to wrap the entire word in braces as shown in the first example.

`\mkpageprefix` [*<pagination>*] [*<postpro>*] {*<text>*}

This command is intended for use in field formatting directives which format the page numbers in the *<postnote>* argument of citation commands and the `pages` field of bibliography entries. It will parse its *<text>* argument and prefix it with ‘p.’ or ‘pp.’ by default. The optional *<pagination>* argument holds the name of a field indicating the pagination type. This may be either `pagination` or `bookpagination`, with `pagination` being the default. The spacing between the prefix and the *<text>* may be modified by redefining `\ppspace`. The default is an unbreakable interword space. See §§ 2.3.10 and 3.12.3 for further details. See also `\DeclareNumChars`, `\DeclareRangeChars`, `\DeclareRangeCommands`, and `\NumCheckSetup`. The optional *<postpro>* argument specifies a macro to be used for post-processing the *<text>*. If only one optional argument is given, it is taken as *<pagination>*. Here are two typical examples:

```
\DeclareFieldFormat{postnote}{\mkpageprefix[pagination]{#1}}
\DeclareFieldFormat{pages}{\mkpageprefix[bookpagination]{#1}}
```

The optional argument `pagination` in the first example is omissible.

`\mkpagetotal` [*<pagination>*] [*<postpro>*] {*<text>*}

This command is similar to `\mkpageprefix` except that it is intended for the `pagetotal` field of bibliography entries, i.e., it will print “123 pages” rather than “page 123”. The optional *<pagination>* argument defaults to `bookpagination`. The spacing inserted between the pagination suffix and the *<text>* may be modified by redefining the macro `\ppspace`. The optional *<postpro>* argument specifies a macro to be used for post-processing the *<text>*. If only one optional argument is given, it is taken as *<pagination>*. Here is a typical example:

```
\DeclareFieldFormat{pagetotal}{\mkpagetotal[bookpagination]{#1}}
```

The optional argument `bookpagination` is omissible in this case.

Input	Output		
	<code>mincomprange=10</code>	<code>mincomprange=100</code>	<code>mincomprange=1000</code>
11-15	11-5	11-15	11-15
111-115	111-5	111-5	111-115
1111-1115	1111-5	1111-5	1111-5
	<code>maxcomprange=1000</code>	<code>maxcomprange=100</code>	<code>maxcomprange=10</code>
1111-1115	1111-5	1111-5	1111-5
1111-1155	1111-55	1111-55	1111-1155
1111-1555	1111-555	1111-1555	1111-1555
	<code>mincompwidth=1</code>	<code>mincompwidth=10</code>	<code>mincompwidth=100</code>
1111-1115	1111-5	1111-15	1111-115
1111-1155	1111-55	1111-55	1111-155
1111-1555	1111-555	1111-555	1111-555

Table 10: `\mkcomprange` setup

`\mkcomprange` [*⟨postpro⟩*] { *⟨text⟩* }

`\mkcomprange*` [*⟨postpro⟩*] { *⟨text⟩* }

This command, which is intended for use in field formatting directives, will parse its *⟨text⟩* argument for page ranges and compress them. For example, “125–129” may be formatted as “125–9”. You may configure the behavior of `\mkcomprange` by adjusting the LaTeX counters `mincomprange`, `maxcomprange`, and `mincompwidth`, as illustrated in table 10. The default settings are 10, 100000, and 1, respectively. This means that the command tries to compress as much as possible by default. Use `\setcounter` to adjust the parameters. The scanner recognizes `\bibrangedash` and hyphens as range dashes. It will normalize the dash by replacing any number of consecutive hyphens with `\bibrangedash`. Lists of ranges delimited with `\bibrangesesep` (Biber³⁸) or commas/semicolons (BibTeX) are also supported. With Biber, the scanner will normalise any comma or semi-colons surrounded by optional space by replacing them with `\bibrangesesep`. If you want to hide a character from the list/range scanner for some reason, wrap the character or the entire string in curly braces. The optional *⟨postpro⟩* argument specifies a macro to be used for post-processing the *⟨text⟩*. This is important if you want to combine `\mkcomprange` with other formatting macros which also need to parse their *⟨text⟩* argument, such as `\mkpageprefix`. Simply nesting these commands will not work as expected. Use the *⟨postpro⟩* argument to set up the processing chain as follows:

```
\DeclareFieldFormat{postnote}{\mkcomprange[{\mkpageprefix[pagination]}]{#1}}
```

Note that `\mkcomprange` is executed first, using `\mkpageprefix` as post-processor. Also note that the *⟨postpro⟩* argument is wrapped in an additional pair of braces. This is only required in this particular case to prevent LaTeX’s optional argument scanner from getting confused by the nested brackets. The starred version of this

³⁸Biber will always convert commas/semicolon multi-range separators into `\bibrangesesep` so that it can be controlled in the style.

command differs from the regular one in the way the $\langle postpro \rangle$ argument is applied to a list of values. For example:

```
\mkcomprange[\mkpageprefix]{5, 123-129, 423-439}  
\mkcomprange*[\mkpageprefix]{5, 123-129, 423-439}
```

will output:

```
pp. 5, 123-9, 423-39  
p. 5, pp. 123-9, pp. 423-39
```

```
\mkfirstpage[\langle postpro \rangle]{\langle text \rangle}  
\mkfirstpage*[\langle postpro \rangle]{\langle text \rangle}
```

This command, which is intended for use in field formatting directives, will parse its $\langle text \rangle$ argument for page ranges and print the start page of the range only. The scanner recognizes `\bibrangedash` and hyphens as range dashes. Lists of ranges delimited with `\bibrangessep` (Biber³⁹) or commas/semicolons (BibTeX) are also supported. If you want to hide a character from the list/range scanner for some reason, wrap the character or the entire string in curly braces. The optional $\langle postpro \rangle$ argument specifies a macro to be used for post-processing the $\langle text \rangle$. See `\mkcomprange` on how to use this argument. The starred version of this command differs from the regular one in the way the $\langle postpro \rangle$ argument is applied to a list of values. For example:

```
\mkfirstpage[\mkpageprefix]{5, 123-129, 423-439}  
\mkfirstpage*[\mkpageprefix]{5, 123-129, 423-439}
```

will output:

```
pp. 5, 123, 423  
p. 5, p. 123, p. 423
```

```
\rangelen{\langle text \rangle}
```

This command will parse its argument as a range and return the length of the range. It will return -1 for open-ended ranges.

```
\rangelen{5-10} returns '5'  
\rangelen{-10} returns '0'  
\rangelen{5-} returns '0'  
\rangelen{5} returns '1'
```

³⁹Biber will always convert commas/semicolon multi-range separators into `\bibrangessep` so that it can be controlled in the style.

This can be used as part of tests in styles which require, for example, ‘f’ as a suffix for ranges of only two pages as when a page range ‘36-37’ should be printed as ‘36f’. This could be done using `\ifnumcomp`:

```
\ifnumcomp{\rangelen{\thefield{pages}}}{=}{1}{add 'f'}{do nothing}
```

`\frangelen{⟨rangefield⟩}`

Biber only

Takes the name of a bibfield declared as a range field in the data model and returns the length of the range. This is calculated by Biber, can handle many special cases and is generally more robust than `\rangelen`. It will return -1 for open ended ranges. Specifically `\frangelen` can:

- Calculate the total of multiple ranges in the same field such as ‘1-10, 20-30’
- Handle implicit ranges such as ‘22-4’ and ‘130-33’
- Handle roman numeral ranges in upper and lower case and consisting of both ASCII and Unicode roman numeral representations.

Here are some examples:

pages = ‘10’	<code>\frangelen{pages}</code> returns ‘1’
pages = ‘10-15’	<code>\frangelen{pages}</code> returns ‘6’
pages = ‘10-15,47-53’	<code>\frangelen{pages}</code> returns ‘13’
pages = ‘10-’	<code>\frangelen{pages}</code> returns ‘-1’
pages = ‘-10’	<code>\frangelen{pages}</code> returns ‘-1’
pages = ‘48-9’	<code>\frangelen{pages}</code> returns ‘2’
pages = ‘172-77’	<code>\frangelen{pages}</code> returns ‘6’
pages = ‘i-vi’	<code>\frangelen{pages}</code> returns ‘6’
pages = ‘X-XX’	<code>\frangelen{pages}</code> returns ‘11’
pages = ‘VII-xii’	<code>\frangelen{pages}</code> returns ‘6’
pages = ‘VII-xii, 145-7, 135-39’	<code>\frangelen{pages}</code> returns ‘14’

As with `\rangelen`, `\frangelen` can be used in tests:

```
\ifnumcomp{\frangelen{pages}}{=}{1}{add 'f'}{do nothing}
```

`\DeclareNumChars{⟨characters⟩}`

`\DeclareNumChars*{⟨characters⟩}`

This command configures the `\ifnumeral`, `\ifnumerals`, and `\ifpages` tests from § 4.6.2. The setup will also affect `\iffieldnum`, `\iffieldnums`, `\iffieldpages` as well as `\mkpageprefix` and `\mkpagetotal`. The `⟨characters⟩` argument is an undelimited list of characters which are to be considered as being part of a number. The regular version of this command replaces the current setting, the starred version appends its argument to the current list. The default setting is:

```
\DeclareNumChars{.}
```

This means that a (section or other) number like ‘3.4.5’ will be considered as a number. Note that Arabic and Roman numerals are detected by default, there is no need to declare them explicitly.

```
\DeclareRangeChars{⟨characters⟩}
```

```
\DeclareRangeChars*{⟨characters⟩}
```

This command configures the `\ifnumerals` and `\ifpages` tests from § 4.6.2. The setup will also affect `\iffieldnums` and `\iffieldpages` as well as `\mkpageprefix` and `\mkpagetotal`. The `⟨characters⟩` argument is an undelimited list of characters which are to be considered as range indicators. The regular version of this command replaces the current setting, the starred version appends its argument to the current list. The default setting is:

```
\DeclareRangeChars{~,; -+ / }
```

This means that strings like ‘3–5’, ‘35+’, ‘8/9’ and so on will be considered as a range by `\ifnumerals` and `\ifpages`. Non-range characters in such strings are recognized as numbers. So strings like ‘3a–5a’ and ‘35b+’ are not deemed to be ranges by default. See also §§ 2.3.10 and 3.12.3 for further details.

```
\DeclareRangeCommands{⟨commands⟩}
```

```
\DeclareRangeCommands*{⟨commands⟩}
```

This command is similar to `\DeclareRangeChars`, except that the `⟨commands⟩` argument is an undelimited list of commands which are to be considered as range indicators. The regular version of this command replaces the current setting, the starred version appends its argument to the current list. The default list is rather long and should cover all common cases; here is a shorter example:

```
\DeclareRangeCommands{\&\bibrangedash\textendash\textemdash\psq\psqq}
```

See also §§ 2.3.10 and 3.12.3 for further details.

```
\DeclarePageCommands{⟨commands⟩}
```

```
\DeclarePageCommands*{⟨commands⟩}
```

This command is similar to `\DeclareRangeCommands`, except that it only affects the `\ifpages` and `\iffieldpages` tests but not `\ifnumerals` and `\iffieldnums`. The default setting is:

```
\DeclarePageCommands{\pno\ppno}
```

`\NumCheckSetup{⟨code⟩}`

Use this command to temporarily redefine any commands which interfere with the tests performed by `\ifnumeral`, `\ifnumerals`, and `\ifpages` from § 4.6.2. The setup will also affect `\iffieldnum`, `\iffieldnums`, `\iffieldpages` as well as `\mkpageprefix` and `\mkpagetotal`. The `⟨code⟩` will be executed in a group by these commands. Since the above mentioned commands will expand the string to be analyzed, it is possible to remove commands to be ignored by the tests by making them expand to an empty string. See also §§ 2.3.10 and 3.12.3 for further details.

`\DeclareCaseLangs{⟨languages⟩}`

`\DeclareCaseLangs*{⟨languages⟩}`

Defines the list of languages which are considered by the `\MakeSentenceCase*` command as it converts a string to sentence case. The `⟨languages⟩` argument is a comma-separated list of babel/polyglossia languages identifiers. The regular version of this command replaces the current setting, the starred version appends its argument to the current list. The default setting is:

```
\DeclareCaseLangs{%  
  american,british,canadian,english,australian,newzealand,  
  USenglish,UKenglish}
```

See the babel/polyglossia manuals and table 2 for a list of languages identifiers.

`\BibliographyWarning{⟨message⟩}`

This command is similar to `\PackageWarning` but prints the entry key of the entry currently being processed in addition to the input line number. It may be used in the bibliography as well as in citation commands. If the `⟨message⟩` is fairly long, use `\MessageBreak` to include line breaks. Note that the standard `\PackageWarning` command does not provide a meaningful clue when used in the bibliography since the input line number is the line on which the `\printbibliography` command was given.

`\RequireBiber[⟨severity⟩]`

This command is intended for use in `cbx/bbx` files and in the `@preamble` of `bib` files. It checks the selected backend and warns if it is not Biber. The optional `⟨severity⟩` argument is an integer specifying the severity. The value 1 triggers an informational message stating that Biber is recommended; 2 triggers a warning stating that Biber is required and the `style/bib` file may not work properly; 3 triggers an error stating that Biber is strictly required and the `style/bib` file will not work at all with any other backend. If `\RequireBiber` is used multiple times, the highest `⟨severity⟩` takes precedence. `cbx/bbx` files on the one hand and the `@preamble` snippets of all `bib` files on the other are tracked separately. If the optional `⟨severity⟩` argument is omitted, the default severity is 2 (warning).

`\pagetrackertrue` These commands activate or deactivate the citation tracker locally (this will affect the
`\pagetrackerfalse` `\iffirstonpage` and `\ifsamepage` test from § 4.6.2). They are intended for use in the definition of citation commands or anywhere in the document body. If a citation command is to be excluded from page tracking, use `\pagetrackerfalse` in the $\langle precode \rangle$ argument of `\DeclareCiteCommand`. See § 4.3.1 for details. Note that these commands have no effect if page tracking has been disabled globally.

`\citetrackertrue` These commands activate or deactivate all citation trackers locally (this will affect the
`\citetrackerfalse` `\ifciteseen`, `\ifentryseen`, `\ifciteibid`, and `\ifciteidem` tests from § 4.6.2). They are intended for use in the definition of citation commands or anywhere in the document body. If a citation command is to be excluded from tracking, use `\citetrackerfalse` in the $\langle precode \rangle$ argument of `\DeclareCiteCommand`. See § 4.3.1 for details. Note that these commands have no effect if tracking has been disabled globally.

`\backtrackertrue` These commands activate or deactivate the `backref` tracker locally. They are intended
`\backtrackerfalse` for use in the definition of citation commands or anywhere in the document body. If a citation command is to be excluded from backtracking, use `\backtrackerfalse` in the $\langle precode \rangle$ argument of `\DeclareCiteCommand`. Note that these commands have no effect if the `backref` option has been not been set globally.

4.7 标点和间距 Punctuation and Spacing

The Biblatex package provides elaborate facilities designed to manage and track punctuation and spacing in the bibliography and in citations. These facilities work on two levels. The high-level commands discussed in § 4.7.1 deal with punctuation and white-space inserted by the bibliography style between the individual segments of a bibliography entry. The commands in §§ 4.7.2, 4.7.3, 4.7.4 work at a lower level. They use TeX's space factor and modified space factor codes to track punctuation in a robust and efficient way. This way it is possible to detect trailing punctuation marks within fields, not only those explicitly inserted between fields. The same technique is also used for automatic capitalization of localization strings, see `\DeclareCapitalPunctuation` in § 4.7.5 as well as § 4.8 for details. Note that these facilities are only made available locally in citations and bibliographies. They will not affect any other part of a document.

4.7.1 块和单元标点 Block and Unit Punctuation

The major segments of a bibliography entry are ‘blocks’ and ‘units’. A block is the larger segment of the two, a unit is shorter or at most equal in length. For example, the values of fields such as `title` or `note` usually form a unit which is separated from subsequent data by a period or a comma. A block may comprise several fields which are treated as separate units, for example `publisher`, `location`, and `year`. The segmentation of an entry into blocks and units is at the discretion of the bibliography style. An entry is segmented by inserting `\newblock` and `\newunit` commands at

suitable places and `\finentry` at the very end (see § 4.2.3 for an example). See also § 4.11.7 for some practical hints.

`\newblock` Records the end of a block. This command does not print anything, it merely marks the end of the block. The block delimiter `\newblockpunct` will be inserted by a subsequent `\printtext`, `\printfield`, `\printlist`, `\printnames`, or `\bibstring` command. You may use `\newblock` at suitable places without having to worry about spurious blocks. A new block will only be started by the next `\printfield` (or similar) command if this command prints anything. See § 4.11.7 for further details.

`\newunit` Records the end of a unit and puts the default delimiter `\newunitpunct` in the punctuation buffer. This command does not print anything, it merely marks the end of the unit. The punctuation buffer will be inserted by the next `\printtext`, `\printfield`, `\printlist`, `\printnames`, or `\bibstring` command. You may use `\newunit` after commands like `\printfield` without having to worry about spurious punctuation and whitespace. The buffer will only be inserted by the next `\printfield` or similar command if *both* fields are non-empty. This also applies to `\printtext`, `\printlist`, `\printnames`, and `\bibstring`. See § 4.11.7 for further details.

`\finentry` Inserts `\finentrypunct`. This command should be used at the very end of every bibliography entry.

`\setunit{⟨punctuation⟩}`

`\setunit*{⟨punctuation⟩}`

The `\setunit` command is similar to `\newunit` except that it uses `⟨punctuation⟩` instead of `\newunitpunct`. The starred variant differs from the regular version in that it checks if the last `\printtext`, `\printfield`, `\printlist`, `\printnames`, or `\bibstring` command did actually print anything. If not, it does nothing.

`\printunit{⟨punctuation⟩}`

`\printunit*{⟨punctuation⟩}`

The `\printunit` command is similar to `\setunit` except that `⟨punctuation⟩` persists in the buffer until the next `\printtext`, `\printfield`, `\printlist`, `\printnames`, or `\bibstring` command.

`\setpunctfont{⟨command⟩}`

This command, which is intended for use in field formatting directives, provides an alternative way of dealing with unit punctuation after a field printed in a different font (for example, a title printed in italics). The standard LaTeX way of dealing with this is adding a small amount of space, the so-called italic correction. This command allows adapting the punctuation to the font of the preceding field. The `⟨command⟩` should be a text font command which takes one argument, such as `\emph` or `\textbf`. This

command will only affect punctuation marks inserted by one of the commands from § 4.7.3. The font adaption is applied to the next punctuation mark only and will be reset automatically thereafter. If you want to reset it manually before it takes effect, issue `\resetpunctfont`. If the `punctfont` package option is disabled, this command does nothing. Note that the `\mkbibemph`, `\mkbibitalic` and `\mkbibbold` wrappers from § 4.10.4 incorporate this feature by default.

`\resetpunctfont` This command resets the unit punctuation font defined with `\setpunctfont` before it takes effect. If the `punctfont` package option is disabled, this command does nothing.

4.7.2 标点判断 Punctuation Tests

The following commands may be used to test for preceding punctuation marks at any point in citations and the bibliography.

`\ifpunct{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if preceded by any punctuation mark except for an abbreviation dot, and `⟨false⟩` otherwise.

`\ifterm{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if preceded by a terminal punctuation mark, and `⟨false⟩` otherwise. A terminal punctuation mark is any punctuation mark which has been registered for automatic capitalization, either with `\DeclareCapitalPunctuation` or by default, see § 4.7.5 for details. By default, this applies to periods, exclamation marks, and question marks.

`\ifpunctmark{⟨character⟩}{⟨true⟩}{⟨false⟩}`

Executes `⟨true⟩` if preceded by the punctuation mark `⟨character⟩`, and `⟨false⟩` otherwise. The `⟨character⟩` may be a comma, a semicolon, a colon, a period, an exclamation mark, a question mark, or an asterisk. Note that a period denotes an end-of-sentence period. Use the asterisk to test for the dot after an abbreviation. If this command is used in a formatting directive for name lists, i. e., in the argument to `\DeclareNameFormat`, the `⟨character⟩` may also be an apostrophe.

4.7.3 添加标点 Adding Punctuation

下面的命令设计用来重复标点。参考文献和标注样式总需要使用这些命令来代替原样输出标点符号。本节中所有的`\add...`命令自动利用`\unspace`命令移除前面的空白 (见 § 4.7.4)。注意: 下面讨论的所有的`\add...`命令的作用是宏包默认的, 无论 Biblatex 换哪种语言都会重新恢复。其作用可以通过`\DeclarePunctuationPairs`命令进行调整, 见 § 4.7.5。节

`\adddot` 如果前面输出的不是任何一种标点符号, 那么添加一个句点 (period)。该命令的目的是在一个缩写后面插入点 (dot)。以这种方式插入的点被认为与其它标点命令插

入的标点性质相同。该命令也用来将前面如实输出的句点 (原样输出的句点, *literal period*) 转换成一个缩写的点。

- `\addcomma` 如果前面输出不是一个逗号 (comma)、分号 (semicolon)、冒号 (colon) 和句点 (period), 那么添加一个逗号。
- `\addsemicolon` Adds a semicolon unless it is preceded by a comma, another semicolon, a colon, or a period.
- `\addcolon` Adds a colon unless it is preceded by a comma, a semicolon, another colon, or a period.
- `\addperiod` 如果前面输出不是一个缩写点或其他任何标点符号, 那么添加一个句号。该命令也可以用来将前面插入的缩写点转换为句号, 比如在句子的末尾⁴⁰。
- `\addexclam` Adds an exclamation mark unless it is preceded by any punctuation mark except for an abbreviation dot.
- `\addquestion` Adds a question mark unless it is preceded by any punctuation mark except for an abbreviation dot.
- `\isdot` 当前面输出的是句号的时候, 将其转换为缩写的点, 如果前面是其它符号那么不添加任何符号。
- `\nopunct` Adds an internal marker which will cause the next punctuation command to print nothing.

4.7.4 添加空格 Adding Whitespace

The following commands are designed to prevent spurious whitespace. Bibliography and citation styles should always use these commands instead of literal whitespace. In contrast to the commands in §§ 4.7.2 and 4.7.3, they are not restricted to citations and the bibliography but available globally.

- `\unspace` Removes preceding whitespace, i. e., removes all skips and penalties from the end of the current horizontal list. This command is implicitly executed by all of the following commands.
- `\addspace` Adds a breakable interword space.
- `\addnbspace` Adds a non-breakable interword space.
- `\addthinspace` Adds a *breakable* thin space.
- `\addnbthinspace` Adds a non-breakable thin space. This is similar to `\,` and `\thinspace`.
- `\addlowpenspace` Adds a space penalized by the value of the `lownamepenalty` counter, see §§ 3.9.3 and 4.10.3 for details.

⁴⁰不是很理解, 前面如果是缩写的点, 那么不加入句号, 那么缩写的点就转变为句号了?

`\addhighpenspace` Adds a space penalized by the value of the `highnamepenalty` counter, see §§ 3.9.3 and 4.10.3 for details.

`\addlpthinspace` Similar to `\addlowpenspace` but adds a breakable thin space.

`\addhpthinspace` Similar to `\addhighpenspace` but adds a breakable thin space.

`\addabbrvspace` Adds a space penalized by the value of the `abbrvpenalty` counter, see §§ 3.9.3 and 4.10.3 for details.

`\addabthinspace` Similar to `\addabbrvspace` but using a thin space.

`\adddotpace` Executes `\adddot` and adds a space penalized by the value of the `abbrvpenalty` counter, see §§ 3.9.3 and 4.10.3 for details.

`\addslash` Adds a breakable slash. This command differs from the `\slash` command in the LaTeX kernel in that a linebreak after the slash is not penalized at all.

Note that the commands in this section implicitly execute `\unspace` to remove spurious whitespace, hence they may be used to override each other. For example, you may use `\addnbspace` to transform a previously inserted interword space into a non-breakable one and `\addspace` to turn a non-breakable space into a breakable one.

41

4.7.5 标点设置和大写 **Configuring Punctuation and Capitalization**

The following commands configure various features related to punctuation and automatic capitalization.⁴²

`\DeclareAutoPunctuation{⟨characters⟩}`

This command defines the punctuation marks to be considered by the citation commands as they scan ahead for punctuation. Note that `⟨characters⟩` is an undelimited list of characters. Valid `⟨characters⟩` are period, comma, semicolon, colon, exclamation and question mark. The default setting is:

```
\DeclareAutoPunctuation{.,;:!?}
```

This definition is restored automatically whenever the `autopunct` package option is set to `true`. Executing `\DeclareAutoPunctuation{}` is equivalent to setting `autopunct=false`, i.e., it disables this feature.

⁴¹注意有的时候`\unspace`看似能够起到作用，但其实并不能随意使用的。在 `beamer` 中 `printtext` 老是有些问题，可能是实现 `printtext` 命令的依赖命令，在 `beamer` 中重定义了，跟 `article` 文档类中的情况差别很大。

⁴²这里的 `capitalization` 是大写的意思么？

`\DeclareCapitalPunctuation{⟨characters⟩}`

When Biblatex inserts localization strings, i. e., key terms such as ‘edition’ or ‘volume’, it automatically capitalizes them after terminal punctuation marks. This command defines the punctuation marks which will cause localization strings to be capitalized if one of them precedes a string. Note that `⟨characters⟩` is an undelimited list of characters. Valid `⟨characters⟩` are period, comma, semicolon, colon, exclamation and question mark. The package default is:

```
\DeclareCapitalPunctuation{.!?}
```

Using `\DeclareCapitalPunctuation` with an empty argument is equivalent to disabling automatic capitalization. Since this feature is language specific, this command must be used in the argument to `\DefineBibliographyExtras` (when used in the preamble) or `\DeclareBibliographyExtras` (when used in a localization module). See §§ 3.8 and 4.9 for details. By default, strings are capitalized after periods, exclamation marks, and question marks. All strings are generally capitalized at the beginning of a paragraph (in fact whenever TeX is in vertical mode).

`\DeclarePunctuationPairs{⟨identifier⟩}{⟨characters⟩}`

Use this command to declare valid pairs of punctuation marks. This will affect the punctuation commands discussed in § 4.7.3. For example, the description of `\addcomma` states that this command adds a comma unless it is preceded by another comma, a semicolon, a colon, or a period. In other words, commas after abbreviation dots, exclamation marks, and question marks are permitted. These valid pairs are declared as follows:

```
\DeclarePunctuationPairs{comma}{*!?!}
```

The `⟨identifier⟩` selects the command to be configured. The identifiers correspond to the names of the punctuation commands from § 4.7.3 without the `\add` prefix, i. e., valid `⟨identifier⟩` strings are `dot`, `comma`, `semicolon`, `colon`, `period`, `exclam`, `question`. The `⟨characters⟩` argument is an undelimited list of punctuation marks. Valid `⟨characters⟩` are comma, semicolon, colon, period, exclamation mark, question mark, and asterisk. A period in the `⟨characters⟩` argument denotes an end-of-sentence period, an asterisk the dot after an abbreviation. This is the default setup, which is automatically restored whenever Biblatex switches languages and corresponds to the behavior described in § 4.7.3:

```
\DeclarePunctuationPairs{dot}{}
\DeclarePunctuationPairs{comma}{*!?!}
\DeclarePunctuationPairs{semicolon}{*!?!}
\DeclarePunctuationPairs{colon}{*!?!}
\DeclarePunctuationPairs{period}{}
\DeclarePunctuationPairs{exclam}{*}
\DeclarePunctuationPairs{question}{*}
```

Since this feature is language specific, `\DeclarePunctuationPairs` must be used in the argument to `\DefineBibliographyExtras` (when used in the preamble) or `\DeclareBibliographyExtras` (when used in a localization module). See §§ 3.8 and 4.9 for details. Note that some localization modules may use a setup which is different from the package default.⁴³

`\DeclareQuotePunctuation{⟨characters⟩}`

This command controls ‘American-style’ punctuation. The `\mkbibquote` wrapper from § 4.10.4 can interact with the punctuation facilities discussed in §§ 4.7.1, 4.7.3, 4.7.4. Punctuation marks after `\mkbibquote` will be moved inside the quotes if they have been registered with `\DeclareQuotePunctuation`. Note that `⟨characters⟩` is an undelimited list of characters. Valid `⟨characters⟩` are period, comma, semicolon, colon, exclamation and question mark. Here is an example:

```
\DeclareQuotePunctuation{.,}
```

Executing `\DeclareQuotePunctuation{}` is equivalent to disabling this feature. This is the package default. Since this feature is language specific, this command must be used in the argument to `\DefineBibliographyExtras` (when used in the preamble) or `\DeclareBibliographyExtras` (when used in a localization module). See §§ 3.8 and 4.9 for details. See also § 3.10.1.

`\uspunctuation` A shorthand using the lower-level commands `\DeclareQuotePunctuation` and `\DeclarePunctuationPairs` to activate ‘American-style’ punctuation. See § 3.10.1 for details. This shorthand is provided for convenience only. The effective settings are applied by the lower-level commands.

`\stdpunctuation` Undoes the settings applied by `\uspunctuation`, restoring standard punctuation. As standard punctuation is the default setting, you only need this command to override a previously executed `\uspunctuation` command. See § 3.10.1 for details.

4.7.6 修正标点追踪 **Correcting Punctuation Tracking**

The facilities for punctuation tracking and automatic capitalization are very reliable under normal circumstances, but there are always marginal cases which may require manual intervention. Typical cases are localization strings printed as the first word in a footnote (which is usually treated as the beginning of a paragraph as far as capitalization is concerned, but TeX is not in vertical mode at this point) or punctuation after periods which are not really end-of-sentence periods (for example, after an ellipsis like “[...]” a command such as `\addperiod` would do nothing since parentheses and brackets are transparent to the punctuation tracker). In such cases, use the following commands in bibliography and citation styles to mark the beginning or middle of a sentence if and where required:

⁴³As of this writing, the `american` module uses different settings for ‘American-style’ punctuation.

- `\bibsentence` This command marks the beginning of a sentence. A localization string immediately after this command will be capitalized and the punctuation tracker is reset, i. e., this command hides all preceding punctuation marks from the punctuation tracker and enforces capitalization.
- `\midsentence` This command marks the middle of a sentence. A localization string immediately after this command will not be capitalized and the punctuation tracker is reset, i. e., this command hides all preceding punctuation marks from the punctuation tracker and suppresses capitalization.
- `\midsentence*` The starred variant of `\midsentence` differs from the regular one in that a preceding abbreviation dot is not hidden from the the punctuation tracker, i. e., any code after `\midsentence*` will see a preceding abbreviation dot. All other punctuation marks are hidden from the punctuation tracker and capitalization is suppressed.

4.8 本地化字符串 Localization Strings

Localization strings are key terms such as ‘edition’ or ‘volume’ which are automatically translated by Biblatex’s localization modules. See § 4.9 for an overview and § 4.9.2 for a list of all strings supported by default. The commands in this section are used to print the localized term.

`\bibstring` [*⟨wrapper⟩*] {*⟨key⟩*}

Prints the localization string *⟨key⟩*, where *⟨key⟩* is an identifier in lowercase letters (see § 4.9.2). The string will be capitalized as required, see § 4.7.5 for details. Depending on the `abbreviate` package option from § 3.1.2.1, `\bibstring` prints the short or the long version of the string. If localization strings are nested, i. e., if `\bibstring` is used in another string, it will behave like `\bibxstring`. If the *⟨wrapper⟩* argument is given, the string is passed to the *⟨wrapper⟩* for formatting. This is intended for font commands such as `\emph`.

`\biblstring` [*⟨wrapper⟩*] {*⟨key⟩*}

Similar to `\bibstring` but always prints the long string, ignoring the `abbreviate` option.

`\bibsstring` [*⟨wrapper⟩*] {*⟨key⟩*}

Similar to `\bibstring` but always prints the short string, ignoring the `abbreviate` option.

`\bibcpstring` [*⟨wrapper⟩*] {*⟨key⟩*}

Similar to `\bibstring` but the term is always capitalized.

`\bibcplstring` [*⟨wrapper⟩*] {*⟨key⟩*}

Similar to `\biblstring` but the term is always capitalized.

`\bibcpsstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\bibsstring` but the term is always capitalized.

`\bibucstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\bibstring` but the whole term is uppercased.

`\bibuclstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\biblstring` but the whole term is uppercased.

`\bibucsstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\bibsstring` but the whole term is uppercased.

`\biblcstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\bibstring` but the whole term is lowercased.

`\biblclstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\biblstring` but the whole term is lowercased.

`\biblcsstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\bibsstring` but the whole term is lowercased.

`\bibxstring{⟨key⟩}`

A simplified but expandable version of `\bibstring`. Note that this variant does not capitalize automatically, nor does it hook into the punctuation tracker. It is intended for special cases in which strings are nested or an expanded localization string is required in a test.

`\bibxlstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\bibxstring` but always uses the long string, ignoring the `abbreviate` option.

`\bibxsstring[⟨wrapper⟩]{⟨key⟩}`

Similar to `\bibxstring` but always uses the short string, ignoring the `abbreviate` option.

`\mainlang`

Switches from the current language to the main document language. This can be used the `⟨wrapper⟩` argument in the localization string commands above.

4.9 本地化模块 Localization Modules

A localization module provides translations for key terms such as ‘edition’ or ‘volume’ as well as definitions for language specific features such as the date format and ordinals. These definitions are provided in files with the suffix `lbx`. The base name of the file must be a language name known to the `babel/polyglossia` packages. The `lbx` files may also be used to map `babel/polyglossia` language names to the backend modules of the `Biblatex` package. All localization modules are loaded on demand in the document body. Note that the contents of the file are processed in a group and that the category code of the character `@` is temporarily set to ‘letter’.

4.9.1 本地化命令 Localization Commands

The user-level versions of the localization commands were already introduced in § 3.8. When used in `lbx` files, however, the syntax of localization commands is different from the user syntax in the preamble and the configuration file. When used in localization files, there is no need to specify the `<language>` because the mapping of strings to a language is already provided by the name of the `lbx` file.

```
\DeclareBibliographyStrings{<definitions>}
```

This command is only available in `lbx` files. It is used to define localization strings. The `<definitions>` consist of `<key>=<value>` pairs which assign an expression to an identifier. A complete list of all keys supported by default is given in § 4.9.2. Note that the syntax of the value is different in `lbx` files. The value assigned to a key consists of two expressions, each of which is wrapped in an additional pair of brackets. This is best shown by example:

```
\DeclareBibliographyStrings{%
  bibliography  = {{Bibliography}{Bibliography}},
  shorthands    = {{List of Abbreviations}{Abbreviations}},
  editor        = {{editor}{ed.}},
  editors       = {{editors}{eds.}},
}
```

The first value is the long, written out expression, the second one is an abbreviated or short form. Both strings must always be given even though they may be identical if an expression is always (or never) abbreviated. Depending on the setting of the `abbreviate` package option (see § 3.1.2.1), `Biblatex` selects one expression when loading the `lbx` file. There is also a special key named `inherit` which copies the strings from a different language. This is intended for languages which only differ in a few expressions, such as German and Austrian or American and British English. For example, here are the complete definitions for Austrian:

```
\DeclareBibliographyStrings{%
```



```
inherit      = {german},
january     = {{J\“anner}{J\“an.}},
}
```

The above examples are slightly simplified. Real localization files should use the punctuation and formatting commands discussed in §§ 4.7.3 and 3.9 instead of literal punctuation. Here is an excerpt from a real localization file:

```
bibliography    = {{Bibliography}{Bibliography}},
shorthands      = {{List of Abbreviations}{Abbreviations}},
editor          = {{editor}{ed\adddot}},
editors         = {{editors}{eds\adddot}},
byeditor        = {{edited by}{ed\adddotsspace by}},
mathesis        = {{Master’s thesis}{MA\addabbrvspace thesis}},
```

Note the handling of abbreviation dots, the spacing in abbreviated expressions, and the capitalization in the example above. All expressions should be capitalized as they usually are when used in the middle of a sentence. The Biblatex package will automatically capitalize the first word when required at the beginning of a sentence, see `\DeclareCapitalPunctuation` in § 4.7.5 for details. Expressions intended for use in headings are special. They should be capitalized in a way that is suitable for titling and should not be abbreviated (but they may have a short form).

`\InheritBibliographyStrings{⟨language⟩}`

This command is only available in `lbx` files. It copies the localization strings for `⟨language⟩` to the current language, as specified by the name of the `lbx` file.

`\DeclareBibliographyExtras{⟨code⟩}`

This command is only available in `lbx` files. It is used to adapt language specific features such as the date format and ordinals. The `⟨code⟩`, which may be arbitrary LaTeX code, will usually consist of redefinitions of the formatting commands from § 4.10.2.

`\UndeclareBibliographyExtras{⟨code⟩}`

This command is only available in `lbx` files. It is used to restore any formatting commands modified with `\DeclareBibliographyExtras`. If a redefined command is included in § 4.10.2, there is no need to restore its previous definition since these commands are localized by all language modules anyway.

`\InheritBibliographyExtras{⟨language⟩}`

This command is only available in `lbx` files. It copies the bibliography extras for `⟨language⟩` to the current language, as specified by the name of the `lbx` file.

`\DeclareHyphenationExceptions{⟨text⟩}`

This command corresponds to `\DefineHyphenationExceptions` from § 3.8. The difference is that it is only available in `lbx` files and that the `⟨language⟩` argument is omitted. The hyphenation exceptions will affect the language of the `lbx` file currently being processed.

`\DeclareRedundantLanguages{⟨language, language, ...⟩}{⟨langid, langid, ...⟩}`

This command provides the language mappings required by the `clearlang` option from § 3.1.2.1. The `⟨language⟩` is the string given in the `language` field (without the optional `lang` prefix); `⟨langid⟩` is babel/polyglossia’s language identifier, as given in the optional argument of `\usepackage` when loading babel or the argument of `\setdefaultlanguage` or `\setotherlanguages` when using polyglossia. This command may be used in `lbx` files or in the document preamble. Here are some examples:

```
\DeclareRedundantLanguages{french}{french}
\DeclareRedundantLanguages{german}{german,ngerman,austrian,naustrian}
\DeclareRedundantLanguages{english,american}{english,american,british,
        canadian,australian,newzealand,USenglish,UKenglish}
```

Note that this feature needs to be enabled globally with the `clearlang` option from § 3.1.2.1. If it is disabled, all mappings will be ignored. If the `⟨langid⟩` parameter is blank, Bibl_{at}ex will clear the mappings for the corresponding `⟨language⟩`, i. e., the feature will be disabled for this `⟨language⟩` only.

`\DeclareLanguageMapping{⟨language⟩}{⟨file⟩}`

This command maps a babel/polyglossia language identifier to an `lbx` file. The `⟨language⟩` must be a language name known to the babel/polyglossia package, i. e., one of the identifiers listed in table 2. The `⟨file⟩` argument is the name of an alternative `lbx` file without the `.lbx` suffix. Declaring the same mapping more than once is possible. Subsequent declarations will simply overwrite any previous ones. This command may only be used in the preamble. See § 4.11.8 for further details.

`\NewBibliographyString{⟨key⟩}`

This command, which may be used in the preamble (including `cbx` and `bbx` files) as well as in `lbx` files, declares new localization strings, i. e., it initializes a new `⟨key⟩` to be used in the `⟨definitions⟩` of `\DefineBibliographyStrings` or `\DeclareBibliographyStrings`. The `⟨key⟩` argument may also be a comma-separated list of key names. When used in an `lbx`, the `⟨key⟩` is initialized only for the language specified by the name of the `lbx` file. The keys listed in § 4.9.2 are defined by default.

4.9.2 本地化关键词 Localization Keys

The localization keys in this section are defined by default and covered by the localization files which come with Biblatex. Note that these strings are only available in citations, the bibliography and bibliography lists. All expressions should be capitalized as they usually are when used in the middle of a sentence. Biblatex will capitalize them automatically at the beginning of a sentence. The only exceptions to these rules are the three strings intended for use in headings.

4.9.2.1 Headings The following strings are special because they are intended for use in headings and made available globally via macros. For this reason, they should be capitalized for use in headings and they must not include any local commands which are part of Biblatex's author interface.

- bibliography** The term 'bibliography', also available as `\bibname`.
- references** The term 'references', also available as `\refname`.
- shorthands** The term 'list of shorthands' or 'list of abbreviations', also available as `\biblistname`.

4.9.2.2 Roles, Expressed as Functions The following keys refer to roles which are expressed as a function ('editor', 'translator') rather than as an action ('edited by', 'translated by').

- editor** The term 'editor', referring to the main editor. This is the most generic editorial role.
- editors** The plural form of `editor`.
- compiler** The term 'compiler', referring to an editor whose task is to compile a work.
- compilers** The plural form of `compiler`.
- founder** The term 'founder', referring to a founding editor.
- founders** The plural form of `founder`.
- continuator** An expression like 'continuator', 'continuation', or 'continued', referring to a past editor who continued the work of the founding editor but was subsequently replaced by the current editor.
- continuators** The plural form of `continuator`.
- redactor** The term 'redactor', referring to a secondary editor.
- redactors** The plural form of `redactor`.
- reviser** The term 'reviser', referring to a secondary editor.
- revisers** The plural form of `reviser`.
- collaborator** A term like 'collaborator', 'collaboration', 'cooperator', or 'cooperation', referring to a secondary editor.
- collaborators** The plural form of `collaborator`.

translator	The term ‘translator’.
translators	The plural form of translator.
commentator	The term ‘commentator’, referring to the author of a commentary to a work.
commentators	The plural form of commentators.
annotator	The term ‘annotator’, referring to the author of annotations to a work.
annotators	The plural form of annotators.

4.9.2.3 Concatenated Editor Roles, Expressed as Functions The following keys are similar in function to `editor`, `translator`, etc. They are used to indicate additional roles of the editor, e. g., ‘editor and translator’, ‘editor and foreword’.

<code>editortr</code>	Used if <code>editor</code> / <code>translator</code> are identical.
<code>editorstr</code>	The plural form of <code>editortr</code> .
<code>editorco</code>	Used if <code>editor</code> / <code>commentator</code> are identical.
<code>editorsco</code>	The plural form of <code>editorco</code> .
<code>editoran</code>	Used if <code>editor</code> / <code>annotator</code> are identical.
<code>editorsan</code>	The plural form of <code>editoran</code> .
<code>editorin</code>	Used if <code>editor</code> / <code>introduction</code> are identical.
<code>editorsin</code>	The plural form of <code>editorin</code> .
<code>editorfo</code>	Used if <code>editor</code> / <code>foreword</code> are identical.
<code>editorsfo</code>	The plural form of <code>editorfo</code> .
<code>editoraf</code>	Used if <code>editor</code> / <code>aftword</code> are identical.
<code>editorsaf</code>	The plural form of <code>editoraf</code> .

Keys for `editor`/`translator`/*role* combinations:

<code>editortrco</code>	Used if <code>editor</code> / <code>translator</code> / <code>commentator</code> are identical.
<code>editorstrco</code>	The plural form of <code>editortrco</code> .
<code>editortran</code>	Used if <code>editor</code> / <code>translator</code> / <code>annotator</code> are identical.
<code>editorstran</code>	The plural form of <code>editortran</code> .
<code>editortrin</code>	Used if <code>editor</code> / <code>translator</code> / <code>introduction</code> are identical.
<code>editorstrin</code>	The plural form of <code>editortrin</code> .
<code>editortrfo</code>	Used if <code>editor</code> / <code>translator</code> / <code>foreword</code> are identical.
<code>editorstrfo</code>	The plural form of <code>editortrfo</code> .
<code>editortraf</code>	Used if <code>editor</code> / <code>translator</code> / <code>aftword</code> are identical.
<code>editorstraf</code>	The plural form of <code>editortraf</code> .

Keys for `editor`/`commentator`/*role* combinations:

`editorcoin` Used if editor/commentator/introduction are identical.

`editorscoin` The plural form of `editorcoin`.

`editorcofo` Used if editor/commentator/foreword are identical.

`editorscofo` The plural form of `editorcofo`.

`editorcoaf` Used if editor/commentator/aftword are identical.

`editorscoaf` The plural form of `editorcoaf`.

Keys for editor/annotator/⟨*role*⟩ combinations:

`editoranin` Used if editor/annotator/introduction are identical.

`editorsanin` The plural form of `editoranin`.

`editoranfo` Used if editor/annotator/foreword are identical.

`editorsanfo` The plural form of `editoranfo`.

`editoranaf` Used if editor/annotator/aftword are identical.

`editorsanaf` The plural form of `editoranaf`.

Keys for editor/translator/commentator/⟨*role*⟩ combinations:

`editortrcoin` Used if editor/translator/commentator/introduction are identical.

`editorstrcoin` The plural form of `editortrcoin`.

`editortrcofo` Used if editor/translator/commentator/foreword are identical.

`editorstrcofo` The plural form of `editortrcofo`.

`editortrcoaf` Used if editor/translator/commentator/aftword are identical.

`editorstrcoaf` The plural form of `editortrcoaf`.

Keys for editor/annotator/commentator/⟨*role*⟩ combinations:

`editortranin` Used if editor/annotator/commentator/introduction are identical.

`editorstranin` The plural form of `editortranin`.

`editortranfo` Used if editor/annotator/commentator/foreword are identical.

`editorstranfo` The plural form of `editortranfo`.

`editortranaf` Used if editor/annotator/commentator/aftword are identical.

`editorstranaf` The plural form of `editortranaf`.

4.9.2.4 Concatenated Translator Roles, Expressed as Functions The following keys are similar in function to `translator`. They are used to indicate additional roles of the translator, e.g., ‘translator and commentator’, ‘translator and introduction’.

`translatorco` Used if `translator`/`commentator` are identical.

<code>translatorsco</code>	The plural form of <code>translatorco</code> .
<code>translatoran</code>	Used if <code>translator/annotator</code> are identical.
<code>translatorsan</code>	The plural form of <code>translatoran</code> .
<code>translatorin</code>	Used if <code>translator/introduction</code> are identical.
<code>translatorsin</code>	The plural form of <code>translatorin</code> .
<code>translatorfo</code>	Used if <code>translator/foreword</code> are identical.
<code>translatorsfo</code>	The plural form of <code>translatorfo</code> .
<code>translatoraf</code>	Used if <code>translator/aftword</code> are identical.
<code>translatorsaf</code>	The plural form of <code>translatoraf</code> .

Keys for `translator/commentator/⟨role⟩` combinations:

<code>translatorcoin</code>	Used if <code>translator/commentator/introduction</code> are identical.
<code>translatorscoin</code>	The plural form of <code>translatorcoin</code> .
<code>translatorcofo</code>	Used if <code>translator/commentator/foreword</code> are identical.
<code>translatorscofo</code>	The plural form of <code>translatorcofo</code> .
<code>translatorcoaf</code>	Used if <code>translator/commentator/aftword</code> are identical.
<code>translatorscoaf</code>	The plural form of <code>translatorcoaf</code> .

Keys for `translator/annotator/⟨role⟩` combinations:

<code>translatoranin</code>	Used if <code>translator/annotator/introduction</code> are identical.
<code>translatorsanin</code>	The plural form of <code>translatoranin</code> .
<code>translatoranfo</code>	Used if <code>translator/annotator/foreword</code> are identical.
<code>translatorsanfo</code>	The plural form of <code>translatoranfo</code> .
<code>translatoranaf</code>	Used if <code>translator/annotator/aftword</code> are identical.
<code>translatorsanaf</code>	The plural form of <code>translatoranaf</code> .

4.9.2.5 Roles, Expressed as Actions The following keys refer to roles which are expressed as an action (‘edited by’, ‘translated by’) rather than as a function (‘editor’, ‘translator’).

<code>byauthor</code>	The expression ‘[created] by <i>⟨name⟩</i> ’.
<code>byeditor</code>	The expression ‘edited by <i>⟨name⟩</i> ’.
<code>bycompiler</code>	The expression ‘compiled by <i>⟨name⟩</i> ’.
<code>byfounder</code>	The expression ‘founded by <i>⟨name⟩</i> ’.
<code>bycontinuator</code>	The expression ‘continued by <i>⟨name⟩</i> ’.
<code>byredactor</code>	The expression ‘redacted by <i>⟨name⟩</i> ’.

<code>byreviser</code>	The expression ‘revised by <i><name></i> ’.
<code>byreviewer</code>	The expression ‘reviewed by <i><name></i> ’.
<code>bycollaborator</code>	An expression like ‘in collaboration with <i><name></i> ’ or ‘in cooperation with <i><name></i> ’.
<code>bytranslator</code>	The expression ‘translated by <i><name></i> ’ or ‘translated from <i><language></i> by <i><name></i> ’.
<code>bycommentator</code>	The expression ‘commented by <i><name></i> ’.
<code>byannotator</code>	The expression ‘annotated by <i><name></i> ’.

4.9.2.6 Concatenated Editor Roles, Expressed as Actions The following keys are similar in function to `byeditor`, `bytranslator`, etc. They are used to indicate additional roles of the editor, e. g., ‘edited and translated by’, ‘edited and furnished with an introduction by’, ‘edited, with a foreword, by’.

<code>byeditortr</code>	Used if <code>editor/translator</code> are identical.
<code>byeditorco</code>	Used if <code>editor/commentator</code> are identical.
<code>byeditoran</code>	Used if <code>editor/annotator</code> are identical.
<code>byeditorin</code>	Used if <code>editor/introduction</code> are identical.
<code>byeditorfo</code>	Used if <code>editor/foreword</code> are identical.
<code>byeditoraf</code>	Used if <code>editor/aftword</code> are identical.

Keys for `editor/translator/<role>` combinations:

<code>byeditortrco</code>	Used if <code>editor/translator/commentator</code> are identical.
<code>byeditortran</code>	Used if <code>editor/translator/annotator</code> are identical.
<code>byeditortrin</code>	Used if <code>editor/translator/introduction</code> are identical.
<code>byeditortrfo</code>	Used if <code>editor/translator/foreword</code> are identical.
<code>byeditortraf</code>	Used if <code>editor/translator/aftword</code> are identical.

Keys for `editor/commentator/<role>` combinations:

<code>byeditorcoin</code>	Used if <code>editor/commentator/introduction</code> are identical.
<code>byeditorcofo</code>	Used if <code>editor/commentator/foreword</code> are identical.
<code>byeditorcoaf</code>	Used if <code>editor/commentator/aftword</code> are identical.

Keys for `editor/annotator/<role>` combinations:

<code>byeditoranin</code>	Used if <code>editor/annotator/introduction</code> are identical.
<code>byeditoranfo</code>	Used if <code>editor/annotator/foreword</code> are identical.
<code>byeditoranaf</code>	Used if <code>editor/annotator/aftword</code> are identical.

Keys for `editor/translator/commentator/<role>` combinations:

- `byeditortrcoin` Used if editor/translator/commentator/introduction are identical.
- `byeditortrcofo` Used if editor/translator/commentator/foreword are identical.
- `byeditortrcoaf` Used if editor/translator/commentator/aftword are identical.

Keys for editor/translator/annotator/⟨*role*⟩ combinations:

- `byeditortranin` Used if editor/annotator/commentator/introduction are identical.
- `byeditortranfo` Used if editor/annotator/commentator/foreword are identical.
- `byeditortranaf` Used if editor/annotator/commentator/aftword are identical.

4.9.2.7 Concatenated Translator Roles, Expressed as Actions The following keys are similar in function to `bytranslator`. They are used to indicate additional roles of the translator, e. g., ‘translated and commented by’, ‘translated and furnished with an introduction by’, ‘translated, with a foreword, by’.

- `bytranslatorco` Used if translator/commentator are identical.
- `bytranslatorsan` Used if translator/annotator are identical.
- `bytranslatorin` Used if translator/introduction are identical.
- `bytranslatorfo` Used if translator/foreword are identical.
- `bytranslatorsaf` Used if translator/aftword are identical.

Keys for translator/commentator/⟨*role*⟩ combinations:

- `bytranslatorcoin` Used if translator/commentator/introduction are identical.
- `bytranslatorcofo` Used if translator/commentator/foreword are identical.
- `bytranslatorcoaf` Used if translator/commentator/aftword are identical.

Keys for translator/annotator/⟨*role*⟩ combinations:

- `bytranslatorsanin` Used if translator/annotator/introduction are identical.
- `bytranslatorsanfo` Used if translator/annotator/foreword are identical.
- `bytranslatorsanaf` Used if translator/annotator/aftword are identical.

4.9.2.8 Roles, Expressed as Objects Roles which are related to supplementary material may also be expressed as objects (‘with a commentary by’) rather than as functions (‘commentator’) or as actions (‘commented by’).

- `withcommentator` The expression ‘with a commentary by ⟨*name*⟩’.
- `withannotator` The expression ‘with annotations by ⟨*name*⟩’.
- `withintroduction` The expression ‘with an introduction by ⟨*name*⟩’.
- `withforeword` The expression ‘with a foreword by ⟨*name*⟩’.
- `withafterword` The expression ‘with an afterword by ⟨*name*⟩’.

4.9.2.9 Supplementary Material

commentary	The term ‘commentary’.
annotations	The term ‘annotations’.
introduction	The term ‘introduction’.
foreword	The term ‘foreword’.
afterword	The term ‘afterword’.

4.9.2.10 Publication Details

volume	The term ‘volume’, referring to a book.
volumes	The plural form of <code>volume</code> .
involumes	The term ‘in’, as used in expressions like ‘in <i><number of volumes></i> volumes’.
jourvol	The term ‘volume’, referring to a journal.
jourser	The term ‘series’, referring to a journal.
book	The term ‘book’, referring to a document division.
part	The term ‘part’, referring to a part of a book or a periodical.
issue	The term ‘issue’, referring to a periodical.
newseries	The expression ‘new series’, referring to a journal.
oldseries	The expression ‘old series’, referring to a journal.
edition	The term ‘edition’.
in	The term ‘in’, referring to the title of a work published as part of another one, e.g., ‘ <i><title of article></i> in <i><title of journal></i> ’.
inseries	The term ‘in’, as used in expressions like ‘volume <i><number></i> in <i><name of series></i> ’.
ofseries	The term ‘of’, as used in expressions like ‘volume <i><number></i> of <i><name of series></i> ’.
number	The term ‘number’, referring to an issue of a journal.
chapter	The term ‘chapter’, referring to a chapter in a book.
version	The term ‘version’, referring to a revision number.
reprint	The term ‘reprint’.
reprintof	The expression ‘reprint of <i><title></i> ’.
reprintas	The expression ‘reprinted as <i><title></i> ’.
reprintfrom	The expression ‘reprinted from <i><title></i> ’.
translationof	The expression ‘translation of <i><title></i> ’.
translationas	The expression ‘translated as <i><title></i> ’.
translationfrom	The expression ‘translated from [the] <i><language></i> ’.
reviewof	The expression ‘review of <i><title></i> ’.

<code>origpubas</code>	The expression ‘originally published as <i><title></i> ’.
<code>origpubin</code>	The expression ‘originally published in <i><year></i> ’.
<code>astitle</code>	The term ‘as’, as used in expressions like ‘published by <i><publisher></i> as <i><title></i> ’.
<code>bypublisher</code>	The term ‘by’, as used in expressions like ‘published by <i><publisher></i> ’.

4.9.2.11 Publication State

<code>inpreparation</code>	The expression ‘in preparation’ (the manuscript is being prepared for publication).
<code>submitted</code>	The expression ‘submitted’ (the manuscript has been submitted to a journal or conference).
<code>forthcoming</code>	The expression ‘forthcoming’ (the manuscript has been accepted by a press or journal).
<code>inpress</code>	The expression ‘in press’ (the manuscript is fully copyedited and out of the author’s hands; it is in the final stages of the production process).
<code>prepublished</code>	The expression ‘pre-published’ (the manuscript is published in a preliminary form or location, such as online version in advance of print publication).

4.9.2.12 Pagination

<code>page</code>	The term ‘page’.
<code>pages</code>	The plural form of <code>page</code> .
<code>column</code>	The term ‘column’, referring to a column on a page.
<code>columns</code>	The plural form of <code>column</code> .
<code>section</code>	The term ‘section’, referring to a document division (usually abbreviated as §).
<code>sections</code>	The plural form of <code>section</code> (usually abbreviated as §§).
<code>paragraph</code>	The term ‘paragraph’ (i. e., a block of text, not to be confused with <code>section</code>).
<code>paragraphs</code>	The plural form of <code>paragraph</code> .
<code>verse</code>	The term ‘verse’ as used when referring to a work which is cited by verse numbers.
<code>verses</code>	The plural form of <code>verse</code> .
<code>line</code>	The term ‘line’ as used when referring to a work which is cited by line numbers.
<code>lines</code>	The plural form of <code>line</code> .

4.9.2.13 Types The following keys are typically used in the `type` field of `@thesis`, `@report`, `@misc`, and other entries:

<code>mathesis</code>	An expression equivalent to the term ‘Master’s thesis’.
<code>phdthesis</code>	The term ‘PhD thesis’, ‘PhD dissertation’, ‘doctoral thesis’, etc.
<code>candthesis</code>	An expression equivalent to the term ‘Candidate thesis’. Used for ‘Candidate’ degrees that have no clear equivalent to the Master’s or doctoral level.

techreport	The term ‘technical report’.
resreport	The term ‘research report’.
software	The term ‘computer software’.
datacd	The term ‘data CD’ or ‘CD-ROM’.
audiocd	The term ‘audio CD’.

4.9.2.14 Miscellaneous

nodate	The term to use in place of a date when there is no date for an entry e. g., ‘n.d.’
and	The term ‘and’, as used in a list of authors or editors, for example.
andothers	The expression ‘and others’ or ‘et alii’, used to mark the truncation of a name list.
andmore	Like <code>andothers</code> but used to mark the truncation of a literal list.

4.9.2.15 Labels The following strings are intended for use as labels, e. g., ‘Address: *<url>*’ or ‘Abstract: *<abstract>*’.

url	The term ‘address’ in the sense of an internet address.
urlfrom	An expression like ‘available from <i><url></i> ’ or ‘available at <i><url></i> ’.
urlseen	An expression like ‘accessed on <i><date></i> ’, ‘retrieved on <i><date></i> ’, ‘visited on <i><date></i> ’, referring to the access date of an online resource.
file	The term ‘file’.
library	The term ‘library’.
abstract	The term ‘abstract’.
annotation	The term ‘annotations’.

4.9.2.16 Citations Traditional scholarly expressions used in citations:

idem	The term equivalent to the Latin ‘idem’ (‘the same [person]’).
idemsf	The feminine singular form of <code>idem</code> .
idemsm	The masculine singular form of <code>idem</code> .
idemsn	The neuter singular form of <code>idem</code> .
idempf	The feminine plural form of <code>idem</code> .
idempm	The masculine plural form of <code>idem</code> .
idempn	The neuter plural form of <code>idem</code> .
idempp	The plural form of <code>idem</code> suitable for a mixed gender list of names.
ibidem	The term equivalent to the Latin ‘ibidem’ (‘in the same place’).
opcit	The term equivalent to the Latin term ‘opere citato’ (‘[in] the work [already] cited’).
loccit	The term equivalent to the Latin term ‘loco citato’ (‘[at] the place [already] cited’).

confer The term equivalent to the Latin ‘confer’ (‘compare’).

sequens The term equivalent to the Latin ‘sequens’ (‘[and] the following [page]’), as used to indicate a range of two pages when only the starting page is provided (e. g., ‘25 sq.’ or ‘25 f.’ instead of ‘25–26’).

sequentes The term equivalent to the Latin ‘sequentes’ (‘[and] the following [pages]’), as used to indicate an open-ended range of pages when only the starting page is provided (e. g., ‘25 sqq.’ or ‘25 ff.’).

passim The term equivalent to the Latin ‘passim’ (‘throughout’, ‘here and there’, ‘scatteredly’).

Other expressions frequently used in citations:

see The term ‘see’.

seealso The expression ‘see also’.

seenote An expression like ‘see note *<footnote>*’ or ‘as in *<footnote>*’, used to refer to a previous footnote in a citation.

backrefpage An expression like ‘see page *<page>*’ or ‘cited on page *<page>*’, used to introduce back references in the bibliography.

backrefpages The plural form of **backrefpage**, e. g., ‘see pages *<pages>*’ or ‘cited on pages *<pages>*’.

quotedin An expression like ‘quoted in *<citation>*’, used when quoting a passage which was already a quotation in the cited work.

citedas An expression like ‘henceforth cited as *<shorthand>*’, used to introduce a shorthand in a citation.

thiscite The expression used in some verbose citation styles to differentiate between the page range of the cited item (typically an article in a journal, collection, or conference proceedings) and the page number the citation refers to. For example: “Author, Title, in: Book, pp. 45–61, **thiscite** p. 52.”

4.9.2.17 Month Names

january The name ‘January’.

february The name ‘February’.

march The name ‘March’.

april The name ‘April’.

may The name ‘May’.

june The name ‘June’.

july The name ‘July’.

august The name ‘August’.

september The name ‘September’.

october	The name ‘October’.
november	The name ‘November’.
december	The name ‘December’.

4.9.2.18 Language Names

langamerican	The language ‘American’ or ‘American English’.
langbrazilian	The language ‘Brazilian’ or ‘Brazilian Portuguese’.
langcatalan	The language ‘Catalan’.
langcroatian	The language ‘Croatian’.
langczech	The language ‘Czech’.
langdanish	The language ‘Danish’.
langdutch	The language ‘Dutch’.
langenglish	The language ‘English’.
langfinnish	The language ‘Finnish’.
langfrench	The language ‘French’.
langgerman	The language ‘German’.
langgreek	The language ‘Greek’.
langitalian	The language ‘Italian’.
langlatin	The language ‘Latin’.
langnorwegian	The language ‘Norwegian’.
langpolish	The language ‘Polish’.
langportuguese	The language ‘Portuguese’.
langrussian	The language ‘Russian’.
langslovene	The language ‘Slovene’.
langspanish	The language ‘Spanish’.
langswedish	The language ‘Swedish’.

The following strings are intended for use in phrases like ‘translated from [the] English by *translator*’:

fromamerican	The expression ‘from [the] American’ or ‘from [the] American English’.
frombrazilian	The expression ‘from [the] Brazilian’ or ‘from [the] Brazilian Portuguese’.
fromcatalan	The expression ‘from [the] Catalan’.
fromczech	The expression ‘from [the] Czech’.
fromcroatian	The expression ‘from [the] Croatian’.
fromdanish	The expression ‘from [the] Danish’.

<code>fromdutch</code>	The expression ‘from [the] Dutch’.
<code>fromenglish</code>	The expression ‘from [the] English’.
<code>fromfinnish</code>	The expression ‘from [the] Finnish’.
<code>fromfrench</code>	The expression ‘from [the] French’.
<code>fromgerman</code>	The expression ‘from [the] German’.
<code>fromgreek</code>	The expression ‘from [the] Greek’.
<code>fromitalian</code>	The expression ‘from [the] Italian’.
<code>fromlatin</code>	The expression ‘from [the] Latin’.
<code>fromnorwegian</code>	The expression ‘from [the] Norwegian’.
<code>frompolish</code>	The expression ‘from [the] Polish’.
<code>fromportuguese</code>	The expression ‘from [the] Portuguese’.
<code>fromrussian</code>	The expression ‘from [the] Russian’.
<code>fromslovene</code>	The expression ‘from [the] Slovene’.
<code>fromspanish</code>	The expression ‘from [the] Spanish’.
<code>fromswedish</code>	The expression ‘from [the] Swedish’.

4.9.2.19 Country Names Country names are localized by using the string `country` plus the iso-3166 country code as the key. The short version of the translation should be the iso-3166 country code. Note that only a small number of country names is defined by default, mainly to illustrate this scheme. These keys are used in the `location` list of `@patent` entries but they may be useful for other purposes as well.

<code>countryde</code>	The name ‘Germany’, abbreviated as DE.
<code>countryeu</code>	The name ‘European Union’, abbreviated as EU.
<code>countryep</code>	Similar to <code>countryeu</code> but abbreviated as EP. This is intended for <code>patent</code> entries.
<code>countryfr</code>	The name ‘France’, abbreviated as FR.
<code>countryuk</code>	The name ‘United Kingdom’, abbreviated (according to ISO-3166) as GB.
<code>countryus</code>	The name ‘United States of America’, abbreviated as US.

4.9.2.20 Patents and Patent Requests Strings related to patents are localized by using the term `patent` plus the iso-3166 country code as the key. Note that only a small number of patent keys is defined by default, mainly to illustrate this scheme. These keys are used in the `type` field of `@patent` entries.

<code>patent</code>	The generic term ‘patent’.
<code>patentde</code>	The expression ‘German patent’.
<code>patenteu</code>	The expression ‘European patent’.

<code>patentfr</code>	The expression ‘French patent’.
<code>patentuk</code>	The expression ‘British patent’.
<code>patentus</code>	The expression ‘U.S. patent’.

Patent requests are handled in a similar way, using the string `patreq` as the base name of the key:

<code>patreq</code>	The generic term ‘patent request’.
<code>patreqde</code>	The expression ‘German patent request’.
<code>patreqeu</code>	The expression ‘European patent request’.
<code>patreqfr</code>	The expression ‘French patent request’.
<code>patrequk</code>	The expression ‘British patent request’.
<code>patrequs</code>	The expression ‘U.S. patent request’.

4.10 格式化命令 Formatting Commands

This section corresponds to § 3.9 in the user part of this manual. Bibliography and citation styles should incorporate the commands and facilities discussed in this section in order to provide a certain degree of high-level configurability. Users should not be forced to write new styles if all they want to do is modify the spacing in the bibliography or the punctuation used in citations.

4.10.1 用户可定义的命令和钩子 User-definable Commands and Hooks

This section corresponds to § 3.9.1 in the user part of the manual. The commands and hooks discussed here are meant to be redefined by users, but bibliography and citation styles may provide a default definition which is different from the package default. These commands are defined in `biblatex.def`. Note that all commands starting with `\mk...` take one mandatory argument.

<code>\bibnamedelima</code>	This delimiter controls the spacing between the elements which make up a name part. It is inserted automatically by the backend after the first name element if the element is less than three characters long and before the last element. The default definition is <code>\addhighpenspace</code> , i. e., a space penalized by the value of the <code>highnamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	Biber only
<code>\bibnamedelimb</code>	This delimiter controls the spacing between the elements which make up a name part. It is inserted automatically by the backend between all name elements where <code>\bibnamedelima</code> does not apply. The default definition is <code>\addlowpenspace</code> , i. e., a space penalized by the value of the <code>lownamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	Biber only

<code>\bibnamedelimc</code>	This delimiter controls the spacing between name parts. The default name formats use it between the name prefix and the last name if <code>useprefix=true</code> . The default definition is <code>\addhighpenspace</code> , i.e., a space penalized by the value of the <code>highnamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	
<code>\bibnamedelimd</code>	This delimiter controls the spacing between name parts. The default name formats use it between all name parts where <code>\bibnamedelimc</code> does not apply. The default definition is <code>\addlowpenspace</code> , i.e., a space penalized by the value of the <code>lownamepenalty</code> counter (§ 3.9.3). Please refer to § 3.12.4 for further details.	
<code>\bibnamedelimi</code>	This delimiter replaces <code>\bibnamedelima/b</code> after initials. Note that this only applies to initials given as such in the <code>bib</code> file, not to the initials automatically generated by Biblatex which use their own set of delimiters.	Biber only
<code>\bibinitperiod</code>	The punctuation inserted automatically by the backend after all initials unless <code>\bibinithyphendelim</code> applies. The default definition is a period (<code>\adddot</code>). Please refer to § 3.12.4 for further details.	Biber only
<code>\bibinitdelim</code>	The spacing inserted automatically by the backend between multiple initials unless <code>\bibinithyphendelim</code> applies. The default definition is an unbreakable interword space. Please refer to § 3.12.4 for further details.	Biber only
<code>\bibinithyphendelim</code>	The punctuation inserted automatically by the backend between the initials of hyphenated name parts, replacing <code>\bibinitperiod</code> and <code>\bibinitdelim</code> . The default definition is a period followed by an unbreakable hyphen. Please refer to § 3.12.4 for further details.	Biber only
<code>\bibindexnamedelima</code>	Replaces <code>\bibnamedelima</code> in the index.	
<code>\bibindexnamedelimb</code>	Replaces <code>\bibnamedelimb</code> in the index.	
<code>\bibindexnamedelimc</code>	Replaces <code>\bibnamedelimc</code> in the index.	
<code>\bibindexnamedelimd</code>	Replaces <code>\bibnamedelimd</code> in the index.	
<code>\bibindexnamedelimi</code>	Replaces <code>\bibnamedelimi</code> in the index.	
<code>\bibindexinitperiod</code>	Replaces <code>\bibinitperiod</code> in the index.	
<code>\bibindexinitdelim</code>	Replaces <code>\bibinitdelim</code> in the index.	
<code>\bibindexinithyphendelim</code>	Replaces <code>\bibinithyphendelim</code> in the index.	
<code>\revsdnamepunct</code>	The punctuation to be printed between the first and last name parts when a name is reversed. The default is a comma. This command should be incorporated in formatting directives for name lists. Please refer to § 3.12.4 for further details.	
<code>\bibnamedash</code>	The dash to be used as a replacement for recurrent authors or editors in the bibliography. The default is an ‘em’ or an ‘en’ dash, depending on the indentation of the list of references.	

- `\labelnamepunct` The separator to be printed after the name used for alphabetizing in the bibliography (author or editor, if the `author` field is undefined). Use this separator instead of `\newunitpunct` at this location. The default is `\newunitpunct`, i. e., it is not handled differently from regular unit punctuation but permits convenient reconfiguration.
- `\subtitlepunct` The separator to be printed between the fields `title` and `subtitle`, `booktitle` and `booksubtitle`, as well as `maintitle` and `mainsubtitle`. Use this separator instead of `\newunitpunct` at this location. The default is `\newunitpunct`, i. e., it is not handled differently from regular unit punctuation but permits convenient reconfiguration.
- `\intitlepunct` The separator to be printed between the word “in” and the following title in entry types such as `@article`, `@inbook`, `@incollection`, etc. Use this separator instead of `\newunitpunct` at this location. The default definition is a colon plus an interword space.
- `\bibpagespunct` The separator to be printed before the `pages` field. Use this separator instead of `\newunitpunct` at this location. The default is a comma plus an interword space.
- `\bibpagerefpunct` The separator to be printed before the `pageref` field. Use this separator instead of `\newunitpunct` at this location. The default is an interword space.
- `\multinamedelim` The delimiter to be printed between multiple items in a name list like `author` or `editor` if there are more than two names in the list. If there are only two names in the list, use the `\finalnamedelim` instead. This command should be incorporated in all formatting directives for name lists.
- `\finalnamedelim` Use this command instead of `\multinamedelim` before the final name in a name list.
- `\revsdnamedelim` The extra delimiter to be printed after the first name in a name list (in addition to `\finalnamedelim`) if the first name is reversed. This command should be incorporated in all formatting directives for name lists.
- `\andothersdelim` The delimiter to be printed before the localization string ‘and others’ if a name list like `author` or `editor` is truncated. This command should be incorporated in all formatting directives for name lists.
- `\multilistdelim` The delimiter to be printed between multiple items in a literal list like `publisher` or `location` if there are more than two names in the list. If there are only two items in the list, use the `\finallistdelim` instead. This command should be incorporated in all formatting directives for literal lists.
- `\finallistdelim` Use this command instead of `\multilistdelim` before the final item in a literal list.
- `\andmoredelim` The delimiter to be printed before the localization string ‘and more’ if a literal list like `publisher` or `location` is truncated. This command should be incorporated in all formatting directives for literal lists.

- `\multicitedelim` The delimiter printed between citations if multiple entry keys are passed to a single citation command. This command should be incorporated in the definition of all citation commands, for example in the $\langle sepcode \rangle$ argument passed to `\DeclareCiteCommand`. See § 4.3.1 for details.
- `\supercitedelim` Similar to `\multinamedelim`, but intended for the `\supercite` command only.
- `\compcitedelim` Similar to `\multicitedelim`, but intended for citation styles that ‘compress’ multiple citations, i.e., print the author only once if subsequent citations share the same author etc.
- `\textcitedelim` Similar to `\multicitedelim`, but intended for `\textcite` and related commands (§ 3.7.2).
- `\nametitledelim` The delimiter to be printed between the author/editor and the title. This command should be incorporated in the definition of all citation commands of author-title and some verbose citation styles.
- `\nameyeardelim` The delimiter to be printed between the author/editor and the year. This command should be incorporated in the definition of all citation commands of author-year citation styles.
- `\volcitedelim` The delimiter to be printed between the volume portion and the page/text portion of `\volcite` and related commands (§ 3.7.6).
- `\prenotedelim` The delimiter to be printed after the $\langle prenote \rangle$ argument of a citation command.
- `\postnotedelim` The delimiter to be printed after the $\langle postnote \rangle$ argument of a citation command.
- `\mkbibnamelast` $\{ \langle text \rangle \}$ Formatting hook for the last name, to be used in all formatting directives for name lists.
- `\mkbibnamefirst` $\{ \langle text \rangle \}$ Similar to `\mkbibnamelast`, but intended for the first name.
- `\mkbibnameprefix` $\{ \langle text \rangle \}$ Similar to `\mkbibnamelast`, but intended for the name prefix.
- `\mkbibnameaffix` $\{ \langle text \rangle \}$ Similar to `\mkbibnamelast`, but intended for the name affix.
- `\relatedpunct` The separator between the `relatedtype` bibliography localization string and the data from the first related entry.
- `\relateddelim` The separator between the data of multiple related entries. The default definition is a linebreak.

4.10.2 不同语言的命令 Language-specific Commands

This section corresponds to § 3.9.2 in the user part of the manual. The commands discussed here are usually handled by the localization modules, but may also be redefined by users on a per-language basis. Note that all commands starting with `\mk...` take one or more mandatory arguments.

`\bibrangedash` The language specific range dash.

`\bibdatedash` The language specific date range dash.

`\mkbibdatelong` Takes the names of three field as arguments which correspond to three date components (in the order year/month/day) and uses their values to print the date in the language specific long date format.

`\mkbibdateshort` Similar to `\mkbibdatelong` but using the language specific short date format.

`\finalandcomma` Prints the comma to be inserted before the final ‘and’ in an enumeration, if applicable in the respective language.

`\finalandsemicolon` Prints the semicolon to be inserted before the final ‘and’ in an enumeration, if applicable in the respective language.

`\mkbibordinal`{*integer*}

Takes an integer argument and prints it as an ordinal number.

`\mkbibmascord`{*integer*}

Similar to `\mkbibordinal`, but prints a masculine ordinal, if applicable in the respective language.

`\mkbibfemord`{*integer*}

Similar to `\mkbibordinal`, but prints a feminine ordinal, if applicable in the respective language.

`\mkbibneutord`{*integer*}

Similar to `\mkbibordinal`, but prints a neuter ordinal, if applicable in the respective language.

`\mkbibordedition`{*integer*}

Similar to `\mkbibordinal`, but intended for use with the term ‘edition’.

`\mkbibordseries`{*integer*}

Similar to `\mkbibordinal`, but intended for use with the term ‘series’.

4.10.3 用户可定义的尺寸和计数器 User-definable Lengths and Counters

This section corresponds to § 3.9.3 in the user part of the manual. The length registers and counters discussed here are meant to be altered by users. Bibliography and citation styles should incorporate them where applicable and may also provide a default setting which is different from the package default.

- `\bibhang` The hanging indentation of the bibliography, if applicable. This length is initialized to `\parindent` at load-time. If `\parindent` is zero length for some reason, `\bibhang` will default to `1em`.
- `\biblabelsep` The horizontal space between entries and their corresponding labels. Bibliography styles which use `list` environments and print a label should set `\labelsep` to `\biblabelsep` in the definition of the respective environment.
- `\bibitemsep` The vertical space between the individual entries in the bibliography. Bibliography styles using `list` environments should set `\itemsep` to `\bibitemsep` in the definition of the respective environment.
- `\bibparsep` The vertical space between paragraphs within an entry in the bibliography. Bibliography styles using `list` environments should set `\parsep` to `\bibparsep` in the definition of the respective environment.
- `abbrvpenalty` The penalty used by `\addabbrvspace`, `\addabthinspace`, and `\adddotsspace`, see § 4.7.4 for details.
- `lownamepenalty` The penalty used by `\addlowpenspace` and `\addlpthinspace`, see § 4.7.4 for details.
- `highnamepenalty` The penalty used by `\addhighpenspace` and `\addhpthinspace`, see § 4.7.4 for details.
- `biburlnumpenalty` If this counter is set to a value greater than zero, Biblatex will permit linebreaks after numbers in all strings formatted with the `\url` command from the `url` package. This will affect URLs and DOIs in the bibliography. The breakpoints will be penalized by the value of this counter. If URLs and/or DOIs in the bibliography run into the margin, try setting this counter to a value greater than zero but less than 10000 (you normally want to use a high value like 9000). Setting the counter to zero disables this feature. This is the default setting.
- `biburlucpenalty` Similar to `biburlnumpenalty`, except that it will add a breakpoint after all uppercase letters.
- `biburllcpenalty` Similar to `biburlnumpenalty`, except that it will add a breakpoint after all lowercase letters.

4.10.4 辅助命令和钩子 Auxiliary Commands and Hooks

The auxiliary commands and facilities in this section serve a special purpose. Some of them are used by Bibl_{at}ex to communicate with bibliography and citation styles in some way or other.

`\mkbibemph{⟨text⟩}`

A generic command which prints its argument as emphasized text. This is a simple wrapper around the standard `\emph` command. Apart from that, it uses `\setpunctfont` from § 4.7.1 to adapt the font of the next punctuation mark following the text set in italics. If the `punctfont` package option is disabled, this command behaves like `\emph`.

`\mkbibitalic{⟨text⟩}`

Similar in concept to `\mkbibemph` but prints italicized text. This is a simple wrapper around the standard `\textit` command which incorporates `\setpunctfont`. If the `punctfont` package option is disabled, this command behaves like `\textit`.

`\mkbibbold{⟨text⟩}`

Similar in concept to `\mkbibemph` but prints bold text. This is a simple wrapper around the standard `\textbf` command which incorporates `\setpunctfont`. If the `punctfont` package option is disabled, this command behaves like `\textbf`.

`\mkbibquote{⟨text⟩}`

A generic command which wraps its argument in quotation marks. If the `csquotes` package is loaded, this command uses the language sensitive quotation marks provided by that package. `\mkbibquote` also supports ‘American-style’ punctuation, see `\DeclareQuotePunctuation` in § 4.7.5 for details.

`\mkbibparens{⟨text⟩}`

A generic command which wraps its argument in parentheses. This command is nestable. When nested, it will alternate between parentheses and brackets, depending on the nesting level.

`\mkbibbrackets{⟨text⟩}`

A generic command which wraps its argument in square brackets. This command is nestable. When nested, it will alternate between brackets and parentheses, depending on the nesting level.

`\bibopenparen⟨text⟩\bibcloseparen`

Alternative syntax for `\mkbibparens`. This will also work across groups. Note that `\bibopenparen` and `\bibcloseparen` must always be balanced.

`\bibopenbracket<text>\bibclosebracket`

Alternative syntax for `\mkbibbrackets`. This will also work across groups. Note that `\bibopenbracket` and `\bibclosebracket` must always be balanced.

`\mkbibfootnote{<text>}`

A generic command which prints its argument as a footnote. This is a wrapper around the standard LaTeX `\footnote` command which removes spurious whitespace preceding the footnote mark and prevents nested footnotes. By default, `\mkbibfootnote` requests capitalization at the beginning of the note and automatically adds a period at the end. You may change this behavior by redefining the `\bibfootnotewrapper` macro introduced below.

`\mkbibfootnotetext{<text>}`

Similar to `\mkbibfootnote` but uses the `\footnotetext` command.

`\mkbibendnote{<text>}`

Similar in concept to `\mkbibfootnote` except that it prints its argument as an endnote. `\mkbibendnote` removes spurious whitespace preceding the endnote mark and prevents nested notes. It supports the `\endnote` command provided by the `endnotes` package as well as the `\pagenote` command provided by the `pagenote` package and the `memoir` class. If both commands are available, `\endnote` takes precedence. If no endnote support is available, `\mkbibendnote` issues an error and falls back to `\footnote`. By default, `\mkbibendnote` requests capitalization at the beginning of the note and automatically adds a period at the end. You may change this behavior by redefining the `\bibendnotewrapper` macro introduced below.

`\mkbibendnotetext{<text>}`

Similar to `\mkbibendnote` but uses the `\endnotetext` command. Please note that as of this writing, neither the `pagenote` package nor the `memoir` class provide a corresponding `\pagenotetext` command. In this case, `\mkbibendnote` will issue an error and fall back to `\footnotetext`.

`\bibfootnotewrapper{<text>}`

An inner wrapper which encloses the `<text>` argument of `\mkbibfootnote` and `\mkbibfootnotetext`. For example, `\mkbibfootnote` eventually boils down to this:

```
\footnote{\bibfootnotewrapper{text}}
```

The wrapper ensures capitalization at the beginning of the note and adds a period at the end. The default definition is:

```
\newcommand{\bibfootnotewrapper}[1]{\bibsentence #1\addperiod}
```

If you don't want capitalization at the beginning or a period at the end of the note, do not modify `\mkbibfootnote` but redefine `\bibfootnotewrapper` instead.

`\bibendnotewrapper{⟨text⟩}`

Similar in concept to `\bibfootnotewrapper` but related to the `\mkbibendnote` and `\mkbibendnotetext` commands.

`\mkbibsuperscript{⟨text⟩}`

A generic command which prints its argument as superscripted text. This is a simple wrapper around the standard LaTeX `\textsuperscript` command which removes spurious whitespace and allows hyphenation of the preceding word.

`\mkbibmonth{⟨integer⟩}`

This command takes an integer argument and prints it as a month name. Even though the output of this command is language specific, its definition is not, hence it is normally not redefined in localization modules.

`\mkdatezeros{⟨integer⟩}`

This command strips leading zeros from a number or preserves them, depending on the `datezeros` package option (§ 3.1.2.1). It is intended for use in the definition of date formatting macros.

`\stripzeros{⟨integer⟩}`

This command strips leading zeros from a number. It is intended for date formatting and ordinals.

`shorthandwidth`

BibTeX only

A special field formatting directive which is used internally by Biblalex. When the bibliographic data is read from the `bbl` file, Biblalex measures the values of all `shorthand` fields and sets the length register `\shorthandwidth` to the width of the widest shorthand (see § 4.10.5). In order to determine the correct width, the package considers two factors: the definition of `\bibfont` and this formatting directive. All styles should adjust this directive such that it corresponds to the format used in the list of shorthands.

``labelfield'width`

Biber only

With Biber, for every field marked as a 'Label field' in the data model, a formatting directive is created as per `shorthandwidth` above. Since `shorthand` is so marked in the default data model, this functionality is a superset of that described for `shorthandwidth`.

`labelnumberwidth` Similar to `shorthandwidth`, but referring to the `labelnumber` field and the length register `\labelnumberwidth`. Numeric styles should adjust this directive such that it corresponds to the format used in the bibliography.

- `labelalphawidth` Similar to `shorthandwidth`, but referring to the `labelalpha` field and the length register `\labelalphawidth`. Alphabetic styles should adjust this directive such that it corresponds to the format used in the bibliography.
- `bibhyperref` A special formatting directive for use with `\printfield` and `\printtext`. This directive wraps its argument in a `\bibhyperref` command, see § 4.6.4 for details.
- `bibhyperlink` A special formatting directive for use with `\printfield` and `\printtext`. It wraps its argument in a `\bibhyperlink` command, see § 4.6.4 for details. The $\langle name \rangle$ argument passed to `\bibhyperlink` is the value of the `entrykey` field.
- `bibhypertarget` A special formatting directive for use with `\printfield` and `\printtext`. It wraps its argument in a `\bibhypertarget` command, see § 4.6.4 for details. The $\langle name \rangle$ argument passed to `\bibhypertarget` is the value of the `entrykey` field.
- `volcitepages` A special formatting directive which controls the format of the page/text portion in the argument of citation commands like `\volcite`.
- `volcitevolume` A special formatting directive which controls the format of the volume portion in the argument of citation commands like `\volcite`.
- `date` A special formatting directive which controls the format of `\printdate` (§ 4.4.1). Note that the date format (long/short etc.) is controlled by the package option `date` from § 3.1.2.1. This formatting directive only controls additional formatting such as fonts etc.
- `datelabel` Similar to `date` but controls the format of `\printdatelabel`.
- `urldate` Similar to `date` but controls the format of `\printurldate`.
- `origdate` Similar to `date` but controls the format of `\printorigdate`.
- `eventdate` Similar to `date` but controls the format of `\printeventdate`.

4.10.5 辅助长度计数器和其它功能 Auxiliary Lengths, Counters, and Other Features

The length registers and counters discussed here are used by Biblatex to pass information to bibliography and citation styles. Think of them as read-only registers. Note that all counters are LaTeX counters. Use `\value{counter}` to read out the current value.

`\shorthandwidth`

BibTeX only

This length register indicates the width of the widest shorthand. Bibliography styles should incorporate this length in the definition of the list of shorthands, if applicable.

With Biber, for every field marked as a ‘Label field’ in the data model, a length register is created as per `shorthandwidth` above. Since `shorthand` is so marked in the default data model, this functionality is a superset of that described for `shorthandwidth`.

`\labelnumberwidth` This length register indicates the width of the widest `labelnumber`. Numeric bibliography styles should incorporate this length in the definition of the bibliography environment.

`\labelalphawidth` This length register indicates the width of the widest `labelalpha`. Alphabetic bibliography styles should incorporate this length in the definition of the bibliography environment.

`maxextraalpha` This counter holds the highest number found in any `extraalpha` field.

`maxextrayear` This counter holds the highest number found in any `extrayear` field.

`refsection` This counter indicates the current `refsection` environment. When queried in a bibliography heading, the counter returns the value of the `refsection` option passed to `\printbibliography`.

`refsegment` This counter indicates the current `refsegment` environment. When queried in a bibliography heading, this counter returns the value of the `refsegment` option passed to `\printbibliography`.

`maxnames` This counter holds the setting of the `maxnames` package option.

`minnames` This counter holds the setting of the `minnames` package option.

`maxitems` This counter holds the setting of the `maxitems` package option.

`minitems` This counter holds the setting of the `minitems` package option.

`instcount` This counter is incremented by Biblatex for every citation as well as for every entry in the bibliography and bibliography lists. The value of this counter uniquely identifies a single instance of a reference in the document.

`citetotal` This counter, which is only available in the `⟨loopcode⟩` of a citation command defined with `\DeclareCiteCommand`, holds the total number of valid entry keys passed to the citation command.

`citecount` This counter, which is only available in the `⟨loopcode⟩` of a citation command defined with `\DeclareCiteCommand`, holds the number of the entry key currently being processed by the `⟨loopcode⟩`.

`multicitetotal` This counter is similar to `citetotal` but only available in `multicite` commands. It holds the total number of citations passed to the `multicite` command. Note that each of these citations may consist of more than one entry key. This information is provided by the `citetotal` counter.

- `multicitecount` This counter is similar to `citecount` but only available in `multicite` commands. It holds the number of the citation currently being processed. Note that this citation may consist of more than one entry key. This information is provided by the `citetotal` and `citecount` counters.
- `listtotal` This counter holds the total number of items in the current list. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else. As an exception, it may also be used in the second optional argument to `\printnames` and `\printlist`, see § 4.4.1 for details. For every list, there is also a counter by the same name which holds the total number of items in the corresponding list. For example, the `author` counter holds the total number of items in the `author` list. This applies to both name lists and literal lists. These counters are similar to `listtotal` except that they may also be used independently of list formatting directives. For example, a bibliography style might check the `editor` counter to decide whether or not to print the term “editor” or rather its plural form “editors” after the list of editors.
- `listcount` This counter holds the number of the list item currently being processed. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else.
- `liststart` This counter holds the `<start>` argument passed to `\printnames` or `\printlist`. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else.
- `liststop` This counter holds the `<stop>` argument passed to `\printnames` or `\printlist`. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else.
- `\currentfield` The name of the field currently being processed by `\printfield`. This information is only available locally in field formatting directives.
- `\currentlist` The name of the literal list currently being processed by `\printlist`. This information is only available locally in list formatting directives.
- `\currentname` The name of the name list currently being processed by `\printnames`. This information is only available locally in name formatting directives.

4.10.6 一般目的的钩子 General Purpose Hooks

`\AtBeginBibliography{<code>}`

Appends the `<code>` to an internal hook executed at the beginning of the bibliography. The `<code>` is executed at the beginning of the list of references, immediately after the `<begin code>` of `\defbibenvironment`. This command may only be used in the preamble.

`\AtBeginShorthands{⟨code⟩}`

Appends the `⟨code⟩` to an internal hook executed at the beginning of the list of shorthands. The `⟨code⟩` is executed at the beginning of the list of shorthands, immediately after the `⟨begin code⟩` of `\defbibenvironment`. This command may only be used in the preamble.

When using Biber, this is just an alias for:

`\AtBeginBiblist{shorthand}{code}`

`\AtBeginBiblist{⟨biblistname⟩}{⟨code⟩}`

Biber only

Appends the `⟨code⟩` to an internal hook executed at the beginning of the bibliography list `⟨biblistname⟩`. The `⟨code⟩` is executed at the beginning of the bibliography list, immediately after the `⟨begin code⟩` of `\defbibenvironment`. This command may only be used in the preamble.

`\AtEveryBibitem{⟨code⟩}`

Appends the `⟨code⟩` to an internal hook executed at the beginning of every item in the bibliography. The `⟨code⟩` is executed immediately after the `⟨item code⟩` of `\defbibenvironment`. The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

`\AtEveryLositem{⟨code⟩}`

Appends the `⟨code⟩` to an internal hook executed at the beginning of every item in the list of shorthands. The `⟨code⟩` is executed immediately after the `⟨item code⟩` of `\defbibenvironment`. The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

When using Biber, this is just an alias for:

`\AtEveryBiblistitem{shorthand}{code}`

`\AtEveryBiblistitem{⟨biblistname⟩}{⟨code⟩}`

Biber only

Appends the `⟨code⟩` to an internal hook executed at the beginning of every item in the bibliography list named `⟨biblistname⟩`. The `⟨code⟩` is executed immediately after the `⟨item code⟩` of `\defbibenvironment`. The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

`\AtNextBibliography{⟨code⟩}`

Similar to `\AtBeginBibliography` but only affecting the next `\printbibliography`. The internal hook is cleared after being executed once. This command may be used in the document body.

`\AtEveryCite{⟨code⟩}`

Appends the `⟨code⟩` to an internal hook executed at the beginning of every citation command. The `⟨code⟩` is executed immediately before the `⟨precode⟩` of the command (see § 4.3.1). No bibliographic data is available at this point. This command may only be used in the preamble.

`\AtEveryCitekey{⟨code⟩}`

Appends the `⟨code⟩` to an internal hook executed once for every entry key passed to a citation command. The `⟨code⟩` is executed immediately before the `⟨loopcode⟩` of the command (see § 4.3.1). The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

`\AtEveryMultiCite{⟨code⟩}`

Appends the `⟨code⟩` to an internal hook executed at the beginning of every multicite command. The `⟨code⟩` is executed immediately before the `multiplenote` field (§ 4.3.2) is printed. No bibliographic data is available at this point. This command may only be used in the preamble.

`\AtNextCite{⟨code⟩}`

Similar to `\AtEveryCite` but only affecting the next citation command. The internal hook is cleared after being executed once. This command may be used in the document body.

`\AtNextCitekey{⟨code⟩}`

Similar to `\AtEveryCitekey` but only affecting the next entry key. The internal hook is cleared after being executed once. This command may be used in the document body.

`\AtNextMultiCite{⟨code⟩}`

Similar to `\AtEveryMultiCite` but only affecting the next multicite command. The internal hook is cleared after being executed once. This command may be used in the document body.

`\AtDataInput[⟨entrytype⟩]{⟨code⟩}`

Appends the `⟨code⟩` to an internal hook executed once for every entry as the bibliographic data is imported from the `bbl` file. The `⟨entrytype⟩` is the entry type the `⟨code⟩` applies to. If it applies to all entry types, omit the optional argument. The `⟨code⟩` is executed immediately after the entry has been imported. This command may only be used in the preamble. Note that `⟨code⟩` may be executed multiple times for an entry. This occurs when the same entry is cited in different `refsection` environments or the `sorting` option settings incorporate more than one sorting scheme. The `refsection` counter holds the number of the respective reference section while the data is imported.

`\UseBibitemHook`

Executes the internal hook corresponding to `\AtEveryBibitem`.

`\UseEveryCiteHook`

Executes the internal hook corresponding to `\AtEveryCite`.

`\UseEveryCitekeyHook`

Executes the internal hook corresponding to `\AtEveryCitekey`.

`\UseEveryMultiCiteHook`

Executes the internal hook corresponding to `\AtMultiEveryCite`.

`\UseNextCiteHook`

Executes and clears the internal hook corresponding to `\AtNextCite`.

`\UseNextCitekeyHook`

Executes and clears the internal hook corresponding to `\AtNextCitekey`.

`\UseNextMultiCiteHook`

Executes and clears the internal hook corresponding to `\AtNextMultiCite`.

`\DeferNextCitekeyHook`

Locally un-defines the internal hook specified by `\AtNextCitekey`. This essentially defers the hook to the next entry key in the citation list, when executed in the *⟨precode⟩* argument of `\DeclareCiteCommand` (§ 4.3.1).

4.11 提示与警告 Hints and Caveats

This section provides some additional hints concerning the author interface of this package. It also addresses common problems and potential misconceptions.

4.11.1 条目集 Entry Sets

Entry sets have already been introduced in § 3.11.5. This section discusses how to process entry sets in a bibliography style. From the perspective of the driver, there is no difference between static and dynamic entry sets. Both types are handled in the same way. You will normally use the `\entryset` command from § 4.4.1 to loop over all set members (in the order in which they are listed in the `entryset` field of the `@set` entry, or in the order in which they were passed to `\defbibentryset`, respectively) and append `\finentry` at the end. That's it. The formatting is handled by the drivers for the entry types of the individual set members:

```
\DeclareBibliographyDriver{set}{%
  \entryset{}}{}%
  \finentry}
```

You may have noticed that the numeric styles which ship with this package support subdivided entry sets, i.e., the members of the set are marked with a letter or some other marker such that citations may either refer to the entire set or to a specific set member. The markers are generated as follows by the bibliography style:

```
\DeclareBibliographyDriver{set}{%
  \entryset
    {\printfield{entrysetcount}%
     \setunit*{\addnbspace}}
    {}%
  \finentry}
```

The `entrysetcount` field holds an integer indicating the position of a set member in the entry set. The conversion of this number to a letter or some other marker is handled by the formatting directive of the `entrysetcount` field. All the driver needs to do is print the field and add some white space (or start a new line). Printing the markers in citations works in a similar way. Where a numeric style normally says `\printfield{labelnumber}`, you simply append the `entrysetcount` field:

```
\printfield{labelnumber}\printfield{entrysetcount}
```

Since this field is only defined when processing citations referring to a set member, there is no need to add any additional tests.

4.11.2 电子出版信息 Electronic Publishing Information

The standard styles feature dedicated support for arXiv references. Support for other resources is easily added. The standard styles handle the `eprint` field as follows:

```
\iffieldundef{eprinttype}
  {\printfield{eprint}}
  {\printfield[eprint:\strfield{eprinttype}]{eprint}}
```

If an `eprinttype` field is available, the above code tries to use the field format `eprint:<eprinttype>`. If this format is undefined, `\printfield` automatically falls back to the field format `eprint`. There are two predefined field formats, the type-specific format `eprint:arxiv` and the fallback format `eprint:`

```
\DeclareFieldFormat{eprint}{...}
\DeclareFieldFormat{eprint:arxiv}{...}
```

In other words, adding support for additional resources is as easy as defining a field format named `eprint:<resource>` where `<resource>` is an identifier to be used in the `eprinttype` field.

4.11.3 额外的摘要和注释 External Abstracts and Annotations

External abstracts and annotations have been discussed in § 3.11.8. This section provides some more background for style authors. The standard styles use the following macros (from `biblatex.def`) to handle abstracts and annotations:

```
\newbibmacro*{annotation}{%
  \iffieldundef{annotation}
    {\printfile[annotation]{\bibannotationprefix\thefield{entrykey}.tex}}%
    {\printfield{annotation}}}%
\newcommand*{\bibannotationprefix}{bibannotation-}

\newbibmacro*{abstract}{%
  \iffieldundef{abstract}
    {\printfile[abstract]{\bibabstractprefix\thefield{entrykey}.tex}}%
    {\printfield{abstract}}}%
\newcommand*{\bibabstractprefix}{bibabstract-}
```

If the `abstract/annotation` field is undefined, the above code tries to load the abstracts/annotations from an external file. The `\printfile` commands also incorporate file name prefixes which may be redefined by users. Note that you must enable `\printfile` explicitly by setting the `loadfiles` package option from § 3.1.2.1. This feature is disabled by default for performance reasons.

4.11.4 消除姓名歧义 Name Disambiguation

Biber only

在 § 3.1.2.3 节引入的 `uniquename` 和 `uniquelist` 选项支持多种操作模式。本节用举例方式介绍不同模式的差别。`uniquename` 选项消除 `labelname` 列表中各姓名间的歧义, `uniquelist` 消除因 `maxnames/minnames` 截短导致的 `labelname` 列表歧义。两个选项可以单独使用也可以联合使用:

4.11.4.1 单个姓名 (姓名内的处理) (`uniquename`) 下面从一些 `uniquename` 例子开始, 考虑如下数据:

```
John Doe    2008
Edward Doe  2008
John Smith  2008
Jane Smith  2008
```

假设我们使用作者年制且设置 `uniquename=false`, 这种情况下, 我们得到如下引用标注:

```
Doe 2008a
Doe 2008b
Smith 2008a
Smith 2008b
```

因为姓有歧义，且所有的年都相同，所以年后附加的字符用来区分并消除歧义。然而，很多样式指南强制要求附加字符只能用于相同作者的区分，而不能用于作者相同的姓的区分。为了消除作者姓的歧义，需要增加姓名的其它完整部分或者缩写来区分。这一需要由`uniquename`选项处理，下面是使用了 `uniquename=init` 的引用标注：

```
J. Doe 2008
E. Doe 2008
Smith 2008a
Smith 2008b
```

`uniquename=init` 限制了用缩写来区分姓名。但因为‘J. Smith’仍然有歧义，所以没有增加。而使用 `uniquename=full`，标注如下：

```
J. Doe 2008
E. Doe 2008
John Smith 2008
Jane Smith 2008
```

为了说明 `uniquename=init/full` 和 `allinit/allfull` 的差别，我们下面介绍 ‘visible’ 姓名的概念。‘visible’ 姓名时位于`maxnames/minnames/uniquelist`截短点前的姓名，比如，给出数据：

```
William Jones/Edward Doe/Jane Smith
John Doe
John Smith
```

`maxnames=1,minnames=1,uniquename=init/full`，我们得到如下的引用标注：

```
Jones et al.
Doe
Smith
```

在消除歧义的时候，`uniquename=init/full` 仅考虑可见的姓名。因为本例中所有的可见姓名的姓都是不同的，所有没有姓名的其他部分附加进来。比较一下使用 `uniquename=allinit` 的输出：

```
Jones et al.
J. Doe
Smith
```

`allinit` 认为所有在`labelname`列表中的姓名，包括列表截短后已经隐藏并且由‘et al.’代替的姓名。在本例中，‘John Doe’与‘Edward Doe’存在歧义。因为两个‘Smiths’无法通过添加缩写的方式区分，所以没有添加。现在来比较一下 `uniquename=allfull` 的输出：

Jones et al.
J. Doe
John Smith

The options `uniquename=mininit/minfull` are similar to `init/full` in that they only consider visible names, but they perform minimal disambiguation. That is, they will disambiguate individual names only if they occur in identical lists of last names. Consider the following data:

`uniquename=mininit/minfull` 选项类似于 `init/full` 仅考虑可见姓名，但仅执行最小的歧义消除。即，仅对姓列表的歧义进行处理，考虑如下数据：

John Doe/William Jones
Edward Doe/William Jones
John Smith/William Edwards
Edward Smith/Allan Johnson

使用 `uniquename=init/full`，得到：

J. Doe and Jones
E. Doe and Jones
J. Smith and Edwards
E. Smith and Johnson

使用 `uniquename=mininit/minfull`，得到：

J. Doe and Jones
E. Doe and Jones
Smith and Edwards
Smith and Johnson

‘Smiths’ 并无歧义，因为姓名列表时没有歧义。`mininit/minfull` 选项仅对姓的列表相同情况进行处理。全局的看姓的列表，注意当未截短的列表的可见名相同的时候，截短的列表时也可能是不同的，比如下面的数据：

John Doe/William Jones
Edward Doe

使用 `maxnames=1,uniquename=init/full`：

J. Doe et al.
E. Doe

使用 `uniquename=mininit/minfull`：

Doe et al.
Doe

因为列表有 ‘et al.’ 的不同，姓名列表就不歧义。

4.11.4.2 姓名列表 (独立姓名外的处理) (uniquelist) 姓名列表也可能存在歧义问题。如果labelname列表由maxnames/minnames选项截短就可能产生歧义。这类问题由uniquelist选处理，考虑如下数据:

```
Doe/Jones/Smith    2005
Smith/Johnson/Doe 2005
Smith/Doe/Edwards 2005
Smith/Doe/Jones    2005
```

很多作者年制样式需要在标注中截短，比如使用 maxnames=1 选项，得到:

```
Doe et al. 2005
Smith et al. 2005a
Smith et al. 2005b
Smith et al. 2005c
```

因为截短后作者存在歧义，所以添加额外字符确保引用标注的唯一性。同样的，一些样式强制要求额外字符只能用于所有作者都相同的情况。为了区分作者列表，必须增加更多的姓名，这样就会超出maxnames/minnames选项设定的截短点。uniquelist选项即描述这一需求，当 uniquelist=true，有:

```
Doe et al. 2005
Smith, Johnson et al. 2005
Smith, Doe and Edwards 2005
Smith, Doe and Jones 2005
```

The uniquelist option overrides maxnames/minnames on a per-entry basis. Essentially, what happens is that the ‘et al.’ part of the citation is expanded to the point of no ambiguity – but no further than that. uniquelist may also be combined with uniqueness. Consider the following data:

uniquelist选项以条目为限重设maxnames/minnames。大体上，标注的‘et al.’部分扩展到无歧义的点-而且也基本到此为止。uniquelist也可以与uniquename联合使用，考虑如下数据:

```
John Doe/Allan Johnson/William Jones 2009
John Doe/Edward Johnson/William Jones 2009
John Doe/Jane Smith/William Jones      2009
John Doe/John Smith/William Jones      2009
John Doe/John Edwards/William Jones    2009
John Doe/John Edwards/Jack Johnson     2009
```

使用 maxnames=1，得到:

```
Doe et al. 2009a
Doe et al. 2009b
```

Doe et al. 2009c
Doe et al. 2009d
Doe et al. 2009e
Doe et al. 2009f

使用 `maxnames=1,uniquename=full,uniquelist=true` 则有:

Doe, A. Johnson et al. 2009
Doe, E. Johnson et al. 2009
Doe, Jane Smith et al. 2009
Doe, John Smith et al. 2009
Doe, Edwards and Jones 2009
Doe, Edwards and Johnson 2009

使用 `uniquelist = minyear`, 消除列表歧义仅在可见列表和`labelyear`相同的时候。这对于仅仅需要整个标注整体具有唯一性的作者年制样式是很有用的, 但是不保证作者姓名的非歧义性。这一模式概念上域`uniquename=mininit/minfull`选项相关。考虑如下例子:

Smith/Jones 2000
Smith/Johnson 2001

使用 `maxnames=1` 和 `uniquelist=true`, 得到:

Smith and Jones 2000
Smith and Johnson 2001

使用 `uniquelist=minyear`, 则得到:

Smith et al. 2000
Smith et al. 2001

使用 `uniquelist=minyear`, 两个文献的作者是否相同并不清楚, 但标注的整体是非歧义的, 因为年份的不同。与此相反, `uniquelist=true` 需要消除作者列表的歧义即便这一信息对于参考文献表的唯一引用是不必要的, 看看如下例子:

Vogel/Beast/Garble/Rook 2000
Vogel/Beast/Tremble/Bite 2000
Vogel/Beast/Acid/Squeeze 2001

使用 `maxnames=3,minnames=1,uniquelist=true`, 得到

Vogel, Beast, Garble et al. 2000
Vogel, Beast, Tremble et al. 2000
Vogel, Beast, Acid et al. 2001

使用 `uniquelist=minyear` 选项，则有：

```
Vogel, Beast, Garble et al. 2000
Vogel, Beast, Tremble et al. 2000
Vogel et al. 2001
```

在最后一个引用中，`uniquelist=minyear` 不重写 `maxnames/minnames`，因为年份的不同，所以不需要消除与其它两个间的歧义。

4.11.5 浮动体和TOC/LOT/LOF中的追踪器 **Trackers in Floats and TOC/LOT/LOF**

If a citation is given in a float (typically in the caption of a figure or table), scholarly back references like ‘ibidem’ or back references based on the page tracker get ambiguous because floats are objects which are (physically and logically) placed outside the flow of text, hence the logic of such references applies poorly to them. To avoid any such ambiguities, the citation and page trackers are temporarily disabled in all floats. In addition to that, these trackers plus the back reference tracker (`backref`) are temporarily disabled in the table of contents, the list of figures, and the list of tables.

4.11.6 混合编程接口 **Mixing Programming Interfaces**

Biblatex 宏包给样式作者提供了 2 个主要的编程接口即：`bbx`文件中使用的`\DeclareBibliographyDriver` 命令用来定义各类参考文献条目的驱动(格式处理器, `handler`)，`cbx`文件中使用的`\DeclareCiteCommand`命令用来定义新的引用命令。然而有时候，混合使用这两个接口会很方便。比如`\fullcite`命令就可以打印类似于完整参考文献条目的长串引用，该命令定义大体如下：

```
\DeclareCiteCommand{\fullcite}
{...}
{\usedriver{...}{\thefield{entrytype}}}
{...}
{...}
```

如上所见，打印引用的核心代码简单地为当前的条目类型执行了`\DeclareBibliographyDriver`定义的参考文献驱动命令。当为长引用样式 (a verbose citation scheme) 编写引用样式文件的时候，使用下面的结构是非常方便的：

```
\ProvidesFile{example.cbx}[2007/06/09 v1.0 biblatex citation style]

\DeclareCiteCommand{\cite}
{...}
{\usedriver{...}{cite:\thefield{entrytype}}}
{...}
```

```
{...}

\DeclareBibliographyDriver{cite:article}{...}
\DeclareBibliographyDriver{cite:book}{...}
\DeclareBibliographyDriver{cite:inbook}{...}
...
```

混合接口的另一个有用情况是在参考文献表中使用交叉引用 (cross-references) 时。比如当打印@incollection 类型的条目，数据继承自@collection父条目，将会由一个简短的指向对应父条目的指针来代替。

[1] Audrey Author: *Title of article*. In: [2], pp. 134–165.

[2] Edward Editor, ed.: *Title of collection*. Publisher: Location, 1995.

实现这种参考文献的交叉引用的一种方法是认为他们是引用关系，使用xref或crossref域的值作为条目关键字 (bibtex 条目键，entry key)，例子如下：

```
\ProvidesFile{example.bbx}[2007/06/09 v1.0 biblatex bibliography style]

\DeclareCiteCommand{\bbx@xref}
{
  {}
  {...}% code for cross-references
  {}
  {}

\DeclareBibliographyDriver{incollection}{%
  ...
  \iffieldundef{xref}
  {...}% code if no cross-reference
  {\bbx@xref{\thefield{xref}}}%
  ...
}
```

当定义\bbx@xref命令时，\DeclareCiteCommand命令的⟨precode⟩，⟨postcode⟩，和⟨sepcode⟩参数留空，是因为上面例子中没有用到。交叉引用由\bbx@xref命令的⟨loopcode⟩参数打印。更多的关于xref域的细节见 § 2.2.3节以及 § 2.4.1节中的注意事项。在 § 4.6.2节我们也看到了\iffieldxref，\iflistxref，和\ifnamexref测试命令。这些都可以用 § 4.4.1节的\entrydata命令来实现。

```
\ProvidesFile{example.bbx}[2007/06/09 v1.0 biblatex bibliography style]
```

```

\DeclareBibliographyDriver{incollection}{%
  ...
  \iffieldundef{xref}
    {...}% code if no cross-reference
    {\entrydata{\thefield{xref}}}%
    % code for cross-references
    ...
  }}%
  ...
}

```

4.11.7 使用标点追踪 Using the Punctuation Tracker

4.11.7.1 标点基础 The Basics 样式作者设计参考文献驱动时需要记住一点: 块和单元的标点是异步处理的。用例子最容易解释这一点, 看下面一段代码:

```

\printfield{title}%
\newunit
\printfield{edition}%
\newunit
\printfield{note}%

```

如果没有edition域, 那么这段代码的打印结果不会是:

Title. . Note

而会是

Title. Note

因为单元的标点追踪器是异步方式工作的。 \newunit命令将不会立即打印标点。它仅是记录一个单元的边界并且将 \newunitpunct命令放入标点缓存中。该缓存会有接下来的 \printfield、 \printlist或类似命令进行处理, 且仅当这些命令各自处理的域或列表已定义的时候才会处理。像 \printfield这样的命令在插入任何块和单元的标点之前将首先考虑 3 个因素:

- 是否有新的单元/块的输出请求?
 - = 前面是否有 \newunit或者 \newblock命令?
- 前面的命令是否有打印输出?
 - = 前面是否有 \printfield或者相似命令?
 - = 该命令是否实际打印了任何东西?
- 现在是否要打印一些东西?
 - 要进行打印处理的域或列表是否已定义?

块和单元的标点只会在上述所有条件满足的时候才会输出。让我们再次考虑上面的例子:

```
\printfield{title}%  
\newunit  
\printfield{edition}%  
\newunit  
\printfield{note}%
```

如果edition域没有定义会发生什么呢? 第一个\printfield命令打印了标题并设置一个内部的‘new text’标志。第一个\newunit命令设置一个内部的‘new unit’标志。这使没有任何标点输出。第二个\printfield命令不进行任何处理因为edition域未定义。接下来的\newunit命令再次设置‘new unit’标志, 仍然没有标点输出。第三个\printfield命令检测note域是否已定义, 如果是, 它会寻找‘new text’和‘new unit’标志。如果两个标志都存在, 那么它会在打印 note 前插入标点缓存。然后它会清除‘new unit’标志然后再次设置‘new text’标志。

所有这些听起来似乎很复杂, 但实际上, 这意味着可以用顺序的方式写一个具有很多部件的参考文献驱动。这种方法的优势在不使用标点追踪而实现上述代码功能时会体现的很明显。如果不用标点追踪, 那么会因为大量对所有可能存在域的判断产生一个复杂的\iffielddundef判断命令集合。

```
\iffielddundef{title}%  
  {\iffielddundef{edition}  
    {\printfield{note}}  
    {\printfield{edition}%  
      \iffielddundef{note}%  
      {}  
      {. \printfield{note}}}}}  
{\printfield{title}%  
  \iffielddundef{edition}  
    {}  
    {. \printfield{edition}}}%  
  \iffielddundef{note}  
    {}  
    {. \printfield{note}}}%
```

4.11.7.2 常见错误 Common Mistakes 把单元的标点处理认为是同步处理的是一个相当常见的误解。这会导致当驱动中包含原样文本⁴⁴时出现一些典型错误。考虑下面导致标点错位的错误代码段:

```
\printfield{title}%
```

⁴⁴这里 literal text 理解为原样文本, 如实文本, 逐字文本

```
\newunit
(\printfield{series} \printfield{number})%
```

这段代码将产生下面的结果:

```
Title (. Series Number)
```

这里发生了什么呢？第一个`\printfield`命令打印了标题，然后`\newunit`命令标记了一个新的单元边界但不打印任何内容。单元的标点由下一个`\printfield`命令打印。这是前面提过的异步机制。然而因为左括号在下一个`\printfield`命令插入标点前立即打印，所以导致了错误的句点。当插入任何原样文本比如括号（还包括由`\bibopenparen`和`\mkbibparens`命令打印的括号）时，总需要将这些文本用`\printtext`命令包起来。要让标点追踪正常运转，需要让驱动知道所有插入的原样文本。这是`\printtext`命令的作用所在。`\printtext`命令联系标点追踪器确保标点缓存在原样文本打印前插入。它也设定内部‘new text’标志。注意本例中还有第三处原样文本即`\printfield{series}`后面的空格。在改正的例子中，我们将使用标点追踪器来处理该空格。

```
\printfield{title}%
\newunit
\printtext{()%
\printfield{series}%
\setunit*{\addspace}%
\printfield{number}%
\printtext{()}%
```

尽管上面的代码能够如常工作，但处理括号、引号和其它包围某个域的标点是，推荐的方式是定义一个域格式:

```
\DeclareFieldFormat{parens}{\mkbibparens{#1}}
```

域格式可以同时用于`\printfield`和`\printtext`命令，因此我们可以利用它对若干个域用一堆括号进行包裹。

```
\printtext[parens]{%
  \printfield{series}%
  \setunit*{\addspace}%
  \printfield{number}%
}%
```

这里我们还需要处理没有 `series` 信息时的情况，因此进一步改进代码如下:

```
\iffielddundef{series}
{}
{\printtext[parens]{%
```



```

\printfield{series}%
\setunit*{\addspace}%
\printfield{number}}}%

```

最后的一点提示：本地化字符串对于标点追踪器来说不是原样文本。因为**\bibstring**和相似命令能联系标点追踪器，因此就不需要用**\printtext**包裹起来。

4.11.7.3 高级应用 Advanced Usage 标点追踪器也可用来处理更复杂的情况。比如，考虑需要对**location**、**publisher**和**year**根据数据是否提供以如下的格式打印：

```

...text. Location: Publisher, Year. Text...
...text. Location: Publisher. Text...
...text. Location: Year. Text...
...text. Publisher, Year. Text...
...text. Location. Text...
...text. Publisher. Text...
...text. Year. Text...

```

这个问题可以用一个相当复杂的**\iflistundef**和**\iffieldundef**判断集进行处理，通过这些判断可以确定所有可能的域的组合：

```

\iflistundef{location}
  {\iflistundef{publisher}
    {\printfield{year}}
    {\printlist{publisher}%
     \iffieldundef{year}
     {}
     {, \printfield{year}}}}
{\printlist{location}%
 \iflistundef{publisher}%
  {\iffieldundef{year}
   {}
   {: \printfield{year}}}
  {: \printlist{publisher}%
   \iffieldundef{year}
   {}
   {, \printfield{year}}}}}%

```

可以应用**\ifthenelse**命令和 § 4.6.3 讨论的布尔运算可使上面的代码更具可读性。但本质上是一样的。然而，也可以按顺序写成如下方式：

```

\newunit
\printlist{location}%

```

```

\setunit*{\addcolon\space}%
\printlist{publisher}%
\setunit*{\addcomma\space}%
\printfield{year}%
\newunit

```

在实际使用中，你会经常使用标点追踪器执行一些显式或隐式的组合判断，比如，考虑如下格式 (注意当没有 `publisher` 时 `location` 后面的标点)

```

...text. Location: Publisher, Year. Text...
...text. Location: Publisher. Text...
...text. Location, Year. Text...
...text. Publisher, Year. Text...
...text. Location. Text...
...text. Publisher. Text...
...text. Year. Text...

```

这可以用如下代码进行处理:

```

\newunit
\printlist{location}%
\iflistundef{publisher}
  {\setunit*{\addcomma\space}}
  {\setunit*{\addcolon\space}}}%
\printlist{publisher}%
\setunit*{\addcomma\space}%
\printfield{year}%
\newunit

```

因为当没有 `publisher` 时 `location` 后面的标点的特殊性，我们需要用一个 `\iflistundef` 判断来确保正确性。剩下其它的则有标点追踪器处理。

4.11.8 定制本地化模型 Custom Localization Modules

Style guides may include provisions as to how strings like ‘edition’ should be abbreviated or they may mandate certain fixed expressions. For example, the MLA style guide requires authors to use the term ‘Works Cited’ rather than ‘Bibliography’ or ‘References’ in the heading of the bibliography. Localization commands such as `\DefineBibliographyStrings` from § 3.8 may indeed be used in `cbx` and `bbx` files to handle such cases. However, overloading style files with translations is rather inconvenient. This is where `\DeclareLanguageMapping` from § 4.9.1 comes into play. This command maps an `lbx` file with alternative translations to a `babel/polyglossia` language. For example, you could create a file named `french-humanities.lbx` which provides French translations adapted for use in the humanities and map it to the `babel/polyglossia` language `french` in the preamble or in the configuration file:

```
\DeclareLanguageMapping{french}{french-humanities}
```

If the document language is set to `french`, `french-humanities.lbx` will replace `french.lbx`. Coming back to the MLA example mentioned above, an MLA style may come with an `american-mla.lbx` file to provide strings which comply with the MLA style guide. It would declare the following mapping in the `cbx` and/or `bbx` file:

```
\DeclareLanguageMapping{american}{american-mla}
```

Since the alternative `lbx` file can inherit strings from the standard `american.lbx` module, `american-mla.lbx` may be as short as this:

```
\ProvidesFile{american-mla.lbx}[2008/10/01 v1.0 biblatex localization]
\InheritBibliographyExtras{american}
\DeclareBibliographyStrings{%
  inherit          = {american},
  bibliography     = {{Works Cited}{Works Cited}},
  references       = {{Works Cited}{Works Cited}},
}
\endinput
```

Alternative `lbx` files must ensure that the localization module is complete. They will typically do so by inheriting data from the corresponding standard module. If the language `american` is mapped to `american-mla.lbx`, Biblatex will not load `american.lbx` unless this module is requested explicitly. In the above example, inheriting ‘strings’ and ‘extras’ will cause Biblatex to load `american.lbx` before applying the modifications in `american-mla.lbx`.

Note that `\DeclareLanguageMapping` is not intended to handle language variants (e.g., American English vs. British English) or `babel/polyglossia` language aliases (e.g., `USenglish` vs. `american`). For example, `babel/polyglossia` offers the `USenglish` option which is similar to `american`. Therefore, Biblatex ships with an `USenglish.lbx` file which simply inherits all data from `american.lbx` (which in turn gets the ‘strings’ from `english.lbx`). In other words, the mapping of language variants and `babel/polyglossia` language aliases happens on the file level, the point being that Biblatex’s language support can be extended simply by adding additional `lbx` files. There is no need for centralized mapping. If you need support for, say, Portuguese (`babel/polyglossia: portuges`), you create a file named `portuges.lbx`. If `babel/polyglossia` offered an alias named `brasil`, you would create `brasil.lbx` and inherit the data from `portuges.lbx`. In contrast to that, the point of `\DeclareLanguageMapping` is handling *stylistic* variants like ‘humanities vs. natural sciences’ or ‘MLA vs. APA’ etc. which will typically be built on top of existing `lbx` files.

4.11.9 编组 Grouping

In a citation or bibliography style, you may need to set flags or store certain values for later use. In this case, it is crucial to understand the basic grouping structure imposed by this package. As a rule of thumb, you are working in a large group whenever author commands such as those discussed in § 4.6 are available because the author interface of this package is only enabled locally. If any bibliographic data is available, there is at least one additional group. Here are some general rules:

- The entire list of references printed by `\printbibliography` and similar commands is processed in a group. Each entry in the list is processed in an additional group which encloses the `\item code` of `\defbibenvironment` as well as all driver code.
- The entire bibliography list printed by `\printbiblist` is processed in a group. Each entry in the list is processed in an additional group which encloses the `\item code` of `\defbibenvironment` as well as all driver code.
- All citation commands defined with `\DeclareCiteCommand` are processed in a group holding the complete citation code consisting of the `\precode`, `\sepcode`, `\loopcode`, and `\postcode` arguments. The `\loopcode` is enclosed in an additional group every time it is executed. If any `\wrapper` code has been specified, the entire unit consisting of the wrapper code and the citation code is wrapped in an additional group.
- In addition to the grouping imposed by all backend commands defined with `\DeclareCiteCommand`, all ‘autocite’ and ‘multicite’ definitions imply an additional group.
- `\printfile`, `\printtext`, `\printfield`, `\printlist`, and `\printnames` form groups. This implies that all formatting directives will be processed within a group of their own.
- All `lbx` files are loaded and processed in a group. If an `lbx` file contains any code which is not part of `\DeclareBibliographyExtras`, the definitions must be global.

Note that using `\aftergroup` in citation and bibliography styles is unreliable because the precise number of groups employed in a certain context may change in future versions of this package. If the above list states that something is processed in a group, this means that there is *at least one* group. There may also be several nested ones.

4.11.10 姓名内的间距 Namespaces

In order to minimize the risk of name clashes, LaTeX packages typically prefix the names of internal macros with a short string specific to the package. For example,

if the `foobar` package requires a macro for internal use, it would typically be called `\FB@macro` or `\foo@macro` rather than `\macro` or `\@macro`. Here is a list of the prefixes used or recommended by Biblatex:

- `blx` All macros with names like `\blx@name` are strictly reserved for internal use. This also applies to counter names, length registers, boolean switches, and so on. These macros may be altered in backwards-incompatible ways, they may be renamed or even removed at any time without further notice. Such changes will not even be mentioned in the revision history or the release notes. In short: never use any macros with the string `blx` in their name in any styles.
- `abx` Macros prefixed with `abx` are also internal macros but they are fairly stable. It is always preferable to use the facilities provided by the official author interface, but there may be cases in which using an `abx` macro is convenient.
- `bbx` This is the recommended prefix for internal macros defined in bibliography styles.
- `cbx` This is the recommended prefix for internal macros defined in citation styles.
- `lbx` This is the recommended base prefix for internal macros defined in localization modules. The localization module should add a second prefix to specify the language. For example, an internal macro defined by the Spanish localization module would be named `\lbx@es@macro`.

附录 Appendix

A 默认的驱动源映射

These are the driver default source mappings. For drivers other than `bibtex` and `ris`, they are highly experimental and subject to change (because the driver datatype itself is unstable or not well suited to bibliographic data).

A.1 `bibtex`

The `bibtex` driver is of course the most comprehensive and mature of the Biblatex/Biber supported data formats. These source mapping defaults are how the aliases from sections § 2.1.2 and § 2.2.5 are implemented.

```
\DeclareDriverSourcemap[datatype=bibtex]{
  \map{
    \step[typesource=conference, typetarget=inproceedings]
    \step[typesource=electronic, typetarget=online]
    \step[typesource=www, typetarget=online]
  }
  \map{
    \step[typesource=mastersthesis, typetarget=thesis, final]
```

```

    \step[fieldset=type,          fieldvalue=mathesis]
  }
  \map{
    \step[typesource=phdthesis, typetarget=thesis, final]
    \step[fieldset=type,          fieldvalue=phdthesis]
  }
  \map{
    \step[typesource=techreport, typetarget=report, final]
    \step[fieldset=type,          fieldvalue=techreport]
  }
  \map{
    \step[fieldsource=address,      fieldtarget=location]
    \step[fieldsource=school,       fieldtarget=institution]
    \step[fieldsource=annote,       fieldtarget=annotation]
    \step[fieldsource=archiveprefix, fieldtarget=eprinttype]
    \step[fieldsource=journal,      fieldtarget=journaltitle]
    \step[fieldsource=primaryclass, fieldtarget=eprintclass]
    \step[fieldsource=key,          fieldtarget=sortkey]
    \step[fieldsource=pdf,          fieldtarget=file]
  }
}

```

A.2 endnotexml

The endnotexml driver for Biber uses the following mappings. This format is quite messy and not well suited to robust typesetting of bibliographic data. It is presented largely as an example of Biber’s modular datasource abilities and also of how to map foreign data models into the Biblatex data model. Mappings from such messy and relatively simple data models are inevitably fragile and are of limited use.

```

\DeclareDriverSourcemap[datatype=endnotexml]{
  \map{
    \step[typesource={Aggregated Database},      typetarget=misc]
    \step[typesource={Ancient Text},             typetarget=misc]
    \step[typesource=Artwork,                    typetarget=artwork]
    \step[typesource={Audiovisual Material},      typetarget=misc]
    \step[typesource=Bill,                       typetarget=jurisdiction]
    \step[typesource=Blog,                      typetarget=online]
    \step[typesource=Book,                      typetarget=book]
    \step[typesource={Book Section},             typetarget=inbook]
    \step[typesource=Case,                      typetarget=jurisdiction]
    \step[typesource=Catalog,                   typetarget=misc]
    \step[typesource={Chart or Table},           typetarget=misc]
  }
}

```

\step[typesource={Classical Work},	typetarget=misc]
\step[typesource={Computer Program},	typetarget=software]
\step[typesource={Conference Paper},	typetarget=inproceedings]
\step[typesource={Conference Proceedings},	typetarget=proceedings]
\step[typesource=Dictionary,	typetarget=inreference]
\step[typesource={Edited Book},	typetarget=collection]
\step[typesource={Electronic Article},	typetarget=article]
\step[typesource={Electronic Book},	typetarget=book]
\step[typesource={Encyclopedia},	typetarget=reference]
\step[typesource=Equation,	typetarget=misc]
\step[typesource=Figure,	typetarget=misc]
\step[typesource={Film or Broadcast},	typetarget=movie]
\step[typesource={Government Document},	typetarget=report]
\step[typesource=Grant,	typetarget=misc]
\step[typesource=Hearing,	typetarget=jurisdiction]
\step[typesource={Journal Article},	typetarget=article]
\step[typesource={Legal Rule or Regulation},	typetarget=legislation]
\step[typesource={Magazine Article},	typetarget=article]
\step[typesource=Manuscript,	typetarget=unpublished]
\step[typesource=Map,	typetarget=misc]
\step[typesource={Newspaper Article},	typetarget=article]
\step[typesource={Online Database},	typetarget=online]
\step[typesource={Online Multimedia},	typetarget=online]
\step[typesource=Pamphlet,	typetarget=booklet]
\step[typesource=Patent,	typetarget=patent]
\step[typesource={Personal Communication},	typetarget=letter]
\step[typesource=Report,	typetarget=report]
\step[typesource=Serial,	typetarget=periodical]
\step[typesource=Standard,	typetarget=standard]
\step[typesource=Statute,	typetarget=legislation]
\step[typesource=Thesis,	typetarget=thesis]
\step[typesource={Unpublished Work},	typetarget=unpublished]
\step[typesource={Web Page},	typetarget=online]
}	
\map{	
\step[fieldsource={electronic-resource-num},	fieldtarget=eprint]
\step[fieldsource={alt-title},	fieldtarget=shorttitle]
\step[fieldsource={meeting-place},	fieldtarget=venue]
\step[fieldsource={pub-location},	fieldtarget=location]
\step[fieldsource={orig-pub},	fieldtarget=origpublisher]
\step[fieldsource={authors},	fieldtarget=author]
\step[fieldsource={secondary-authors},	fieldtarget=editor]

```

\step[fieldsource={tertiary-authors}, fieldtarget=commentator]
\step[fieldsource={subsidiary-authors}, fieldtarget=translator]
\step[fieldsource={year}, fieldtarget=date]
\step[fieldsource={pub-dates}, fieldtarget=date]
\step[fieldsource={num-vols}, fieldtarget=volumes]
\step[fieldsource={call-num}, fieldtarget=library]
\step[fieldsource={notes}, fieldtarget=note]
\step[fieldsource={secondary-title}, fieldtarget=subtitle]
\step[fieldsource={work-type}, fieldtarget=type]
}
\map{
  \pertype{Edited Book}
  \step[fieldsource=contributors/authors, fieldtarget=contributors/editor]
}
\map{
  \pertype{Electronic Article}
  \pertype{Journal Article}
  \pertype{Magazine Article}
  \pertype{Newspaper Article}
  \step[fieldsource=isbn, fieldtarget=issn]
}
\map{
  \pertype{Patent}
  \pertype{Report}
  \pertype{Government Document}
  \pertype{Legal Rule or Regulation}
  \step[fieldsource=isbn, fieldtarget=number]
}
\map{
  \pertype{Blog}
  \pertype{Online Database}
  \pertype{Online Multimedia}
  \pertype{Web Page}
  \step[fieldsource={titles/secondary-title}, fieldtarget={titles/title}]
}
\map{
  \pertype{Book Section}
  \step[fieldsource={titles/secondary-title}, fieldtarget={titles/booktitle}]
}
\map{
  \pertype{Book}
  \pertype{Electronic Book}

```



```

\pertype{Manuscript}
\pertype{Unpublished Work}
\step[fieldsource={titles/secondary-title}, fieldtarget={titles/series}]
}
\map{
\pertype{Conference Paper}
\pertype{Conference Proceedings}
\step[fieldsource={titles/secondary-title}, fieldtarget={titles/eventtitle}]
}
\map{
\pertype{Electronic Article}
\pertype{Journal Article}
\pertype{Magazine Article}
\pertype{Newspaper Article}
\step[fieldsource={titles/secondary-title}, fieldtarget={titles/journaltitle}]
}
\map{
\pertype{Book Section}
\step[fieldsource={titles/tertiary-title}, fieldtarget={titles/booktitle}]
}
\map{
\pertype{Conference Proceedings}
\pertype{periodical}
\step[fieldsource={titles/tertiary-title}, fieldtarget={titles/series}]
}
}

```

A.3 ris

The `ris` driver reflects the fact that `ris` itself is a very simple and stable format. It is in fact so simple, it's hardly of any use for most Biblatex users. Again, here more as a proof of concept example.

```

\DeclareDriverSourcemap[datatype=ris]{
\map{
\step[typesource=ART,      typetarget=artwork]
\step[typesource=BILL,    typetarget=jurisdiction]
\step[typesource=BOOK,    typetarget=book]
\step[typesource=CHAP,    typetarget=inbook]
\step[typesource=COMP,    typetarget=software]
\step[typesource=CONF,    typetarget=proceedings]
\step[typesource=GEN,     typetarget=misc]
\step[typesource=JFULL,   typetarget=article]
}
}

```

```

\step[typesource=JOUR,      typetarget=article]
\step[typesource=MGZN,      typetarget=misc]
\step[typesource=MPCT,      typetarget=movie]
\step[typesource=NEWS,      typetarget=misc]
\step[typesource=PAMP,      typetarget=misc]
\step[typesource=PAT,        typetarget=patent]
\step[typesource=PCOMM,      typetarget=misc]
\step[typesource=RPRT,      typetarget=report]
\step[typesource=SER,        typetarget=misc]
\step[typesource=SLIDE,      typetarget=misc]
\step[typesource=SOUND,      typetarget=audio]
\step[typesource=STAT,      typetarget=legal]
\step[typesource=THES,      typetarget=thesis]
\step[typesource=UNBILL,     typetarget=jurisdiction]
\step[typesource=UNPB,      typetarget=unpublished]
}
\map{
  \step[fieldsource=Y1,      fieldtarget=date]
  \step[fieldsource=PY,      fieldtarget=date]
  \step[fieldsource=Y2,      fieldtarget=eventdate]
  \step[fieldsource=A1,      fieldtarget=author]
  \step[fieldsource=AU,      fieldtarget=author]
  \step[fieldsource=A2,      fieldtarget=editor]
  \step[fieldsource=A3,      fieldtarget=editor]
  \step[fieldsource=ED,      fieldtarget=editor]
  \step[fieldsource=SPEP,     fieldtarget=pages]
  \step[fieldsource=N1,      fieldtarget=note]
  \step[fieldsource=N2,      fieldtarget=abstract]
  \step[fieldsource=AB,      fieldtarget=abstract]
  \step[fieldsource=JO,      fieldtarget=journaltitle]
  \step[fieldsource=JF,      fieldtarget=journaltitle]
  \step[fieldsource=JA,      fieldtarget=shortjournal]
  \step[fieldsource=VL,      fieldtarget=volume]
  \step[fieldsource=IS,      fieldtarget=issue]
  \step[fieldsource=CP,      fieldtarget=issue]
  \step[fieldsource=CY,      fieldtarget=location]
  \step[fieldsource=SN,      fieldtarget=isbn]
  \step[fieldsource=PB,      fieldtarget=publisher]
  \step[fieldsource=KW,      fieldtarget=keywords]
  \step[fieldsource=TI,      fieldtarget=title]
  \step[fieldsource=U1,      fieldtarget=usera]
  \step[fieldsource=U2,      fieldtarget=userb]

```

```

\step[fieldsource=U3,      fieldtarget=userc]
\step[fieldsource=U4,      fieldtarget=userd]
\step[fieldsource=U5,      fieldtarget=usere]
\step[fieldsource=UR,      fieldtarget=url]
\step[fieldsource=L1,      fieldtarget=file]
}
}

```

A.4 zoterordfxml

The `zoterordfxml` driver reads the Zotero⁴⁵ XML representation of its RDF format. From the Biblatex point of view, it suffers from a lack of the notion of a citation key and so is of limited use. The format may include a citation key in the future at which point the Biber driver and these default mappings will be updated. As with `endnotexml`, the format is somewhat messy with, in particular, a very limited way of dealing with related entries which makes the driver setup rather ugly. As can be seen below, it is possible to map into a field prefixed with “BIBERCUSTOM” which is not a real field in the data model but can be detected in the Biber driver itself and used to process a source field which does not map naturally to any Biblatex data model field as it needs special treatment. This is messy but is due largely due to messy source data models.

```

\DeclareDriverSourcemap[datatype=zoterordfxml]{
  \map{
    \step[typesource=conferencePaper,    typetarget=inproceedings]
    \step[typesource=bookSection,        typetarget=inbook]
    \step[typesource=journalArticle,      typetarget=article]
    \step[typesource=magazineArticle,     typetarget=article]
    \step[typesource=newspaperArticle,    typetarget=article]
    \step[typesource=encyclopediaArticle, typetarget=inreference]
    \step[typesource=manuscript,          typetarget=unpublished]
    \step[typesource=document,            typetarget=misc]
    \step[typesource=dictionaryEntry,     typetarget=inreference]
    \step[typesource=interview,           typetarget=misc]
    \step[typesource=film,                typetarget=movie]
    \step[typesource=webpage,             typetarget=online]
    \step[typesource=note,                typetarget=misc]
    \step[typesource=attachment,          typetarget=misc]
    \step[typesource=bill,                typetarget=legislation]
    \step[typesource=case,                typetarget=jurisdiction]
    \step[typesource=hearing,             typetarget=jurisdiction]
    \step[typesource=statute,             typetarget=legislation]
    \step[typesource=email,               typetarget=letter]
  }
}

```

⁴⁵www.zotero.org

```

\step[typesource=map,                typetarget=image]
\step[typesource=blogPost,           typetarget=online]
\step[typesource=instantMessage,     typetarget=letter]
\step[typesource=forumPost,          typetarget=online]
\step[typesource=audioRecording,      typetarget=audio]
\step[typesource=presentation,        typetarget=inproceedings]
\step[typesource=videoRecording,      typetarget=video]
\step[typesource=tvBroadcast,         typetarget=misc]
\step[typesource=radioBroadcast,      typetarget=misc]
\step[typesource=podcast,             typetarget=online]
\step[typesource=computerProgram,     typetarget=software]
}
\map{
  \step[fieldsource=bib:contributors, fieldtarget=author]
  \step[fieldsource=bib:authors,      fieldtarget=author]
  \step[fieldsource=z:interviewers,    fieldtarget=author]
  \step[fieldsource=z:directors,       fieldtarget=author]
  \step[fieldsource=z:scriptwriters,   fieldtarget=author]
  \step[fieldsource=z:bookAuthor,      fieldtarget=author]
  \step[fieldsource=z:inventors,       fieldtarget=author]
  \step[fieldsource=z:recipients,      fieldtarget=author]
  \step[fieldsource=z:counsels,        fieldtarget=author]
  \step[fieldsource=z:artists,         fieldtarget=author]
  \step[fieldsource=z:podcasters,      fieldtarget=author]
  \step[fieldsource=z:presenters,      fieldtarget=author]
  \step[fieldsource=z:commenters,      fieldtarget=author]
  \step[fieldsource=z:programers,      fieldtarget=author]
  \step[fieldsource=z:composers,       fieldtarget=author]
  \step[fieldsource=z:producers,       fieldtarget=author]
  \step[fieldsource=z:performers,      fieldtarget=author]
  \step[fieldsource=bib:editors,       fieldtarget=editor]
  \step[fieldsource=z:translators,     fieldtarget=translator]
  \step[fieldsource=z:seriesEditors,   fieldtarget=editor]
  \step[fieldsource=dc:date,           fieldtarget=date]
  \step[fieldsource=bib:pages,         fieldtarget=pages]
  \step[fieldsource=dc:title,          fieldtarget=title]
  \step[fieldsource=z:proceedingsTitle, fieldtarget=title]
  \step[fieldsource=z:encyclopediaTitle, fieldtarget=title]
  \step[fieldsource=z:dictionaryTitle, fieldtarget=title]
  \step[fieldsource=z:websiteTitle,    fieldtarget=title]
  \step[fieldsource=z:forumTitle,      fieldtarget=title]
  \step[fieldsource=z:blogTitle,       fieldtarget=title]

```

```

\step[fieldsource=z:nameOfAct,          fieldtarget=title]
\step[fieldsource=z:caseName,          fieldtarget=title]
\step[fieldsource=z:meetingName,       fieldtarget=eventtitle]
\step[fieldsource=prism:volume,        fieldtarget=volume]
\step[fieldsource=numberOfVolumes,     fieldtarget=volumes]
\step[fieldsource=z:numPages,          fieldtarget=pagetotal]
\step[fieldsource=prism:edition,       fieldtarget=edition]
\step[fieldsource=dc:description,      fieldtarget=note]
\step[fieldsource=dc:alternative,      fieldtarget=shortjournal]
\step[fieldsource=dcterms:abstract,    fieldtarget=abstract]
\step[fieldsource=dc:type,            fieldtarget=type]
\step[fieldsource=z:shortTitle,        fieldtarget=shorttitle]
\step[fieldsource=z:bookTitle,         fieldtarget=booktitle]
\step[fieldsource=prism:number,        fieldtarget=number]
\step[fieldsource=z:patentNumber,      fieldtarget=number]
\step[fieldsource=z:codeNumber,        fieldtarget=number]
\step[fieldsource=z:reportNumber,      fieldtarget=number]
\step[fieldsource=z:billNumber,        fieldtarget=number]
\step[fieldsource=z:documentNumber,    fieldtarget=number]
\step[fieldsource=z:publicLawNumber,   fieldtarget=number]
\step[fieldsource=z:applicationNumber, fieldtarget=number]
\step[fieldsource=z:episodeNumber,     fieldtarget=number]
\step[fieldsource=dc:coverage,         fieldtarget=location]
\step[fieldsource=z:university,        fieldtarget=institution]
\step[fieldsource=z:language,         fieldtarget=language]
\step[fieldsource=z:version,          fieldtarget=version]
\step[fieldsource=z:libraryCatalog,    fieldtarget=library]
\step[fieldsource=dcterms:isPartOf,    fieldtarget=BIBERCUSTOMpartof]
\step[fieldsource=dc:identifier,       fieldtarget=BIBERCUSTOMidentifier]
\step[fieldsource=dc:publisher,        fieldtarget=BIBERCUSTOMpublisher]
\step[fieldsource=dc:presentedAt,      fieldtarget=BIBERCUSTOMpresentedat]
\step[fieldsource=dc:subject,          fieldtarget=BIBERCUSTOMsubject]
\step[fieldsource={dcterms:BIBERCUSTOMpartof/bib:Journal},
      fieldtarget={dcterms:BIBERCUSTOMpartof/periodical}]
\step[fieldsource={dcterms:BIBERCUSTOMpartof/bib:Book},
      fieldtarget={dcterms:BIBERCUSTOMpartof/book}]
\step[fieldsource={dcterms:BIBERCUSTOMpartof/bib:ConferenceProceedings},
      fieldtarget={dcterms:BIBERCUSTOMpartof/proceedings}]
}
}

```

B 默认的继承设置

The following table shows the Biber cross-referencing rules defined by default. Please refer to §§ 2.4.1 and 4.5.10 for explanation.

Types		Fields	
Source	Target	Source	Target
*	*	ids	-
		crossref	-
		xref	-
		entryset	-
		entrysubtype	-
		execute	-
		label	-
		options	-
		presort	-
		related	-
		relatedoptions	-
		relatedstring	-
		relatedtype	-
		shorthand	-
		shorthandintro	-
		sortkey	-
mvbook, book	inbook, bookinbook, suppbook	author	author
		author	bookauthor
mvbook	book, inbook, bookinbook, suppbook	title	maintitle
		subtitle	mainsubtitle
		titleaddon	maintitleaddon
		shorttitle	-
		sorttitle	-
		indextitle	-
		indexsorttitle	-
		title	maintitle
mvcollection, mvreference	collection, reference, incollection, inreference, suppcollection	subtitle	mainsubtitle
		titleaddon	maintitleaddon
		shorttitle	-
		sorttitle	-
		indextitle	-
		indexsorttitle	-
		title	maintitle
		subtitle	mainsubtitle
mvproceedings	proceedings, inproceedings	titleaddon	maintitleaddon
		shorttitle	-
		sorttitle	-
		indextitle	-
		indexsorttitle	-
		title	booktitle
		subtitle	booksubtitle
		titleaddon	booktitleaddon
book	inbook, bookinbook, suppbook	shorttitle	-
		sorttitle	-
		indextitle	-
		indexsorttitle	-
		title	booktitle

Types		Fields	
Source	Target	Source	Target
collection, reference	incollection, inreference, suppcollection	title	booktitle
		subtitle	booksubtitle
		titleaddon	booktitleaddon
		shorttitle	-
		sorttitle	-
		indextitle	-
proceedings	inproceedings	indexsorttitle	-
		title	booktitle
		subtitle	booksubtitle
		titleaddon	booktitleaddon
		shorttitle	-
		sorttitle	-
periodical	article, suppperiodical	indextitle	-
		indexsorttitle	-
		title	journaltitle
		subtitle	journalsubtitle
		shorttitle	-
		sorttitle	-
		indextitle	-
		indexsorttitle	-

C 默认的排序方式

C.1 Alphabetic Schemes 1

The following table shows the standard alphabetic sorting schemes defined by default. Please refer to § 3.5 for explanation.

Option	Sorting scheme				
nty	presort	→ sortname	→ sorttitle	→ sortyear	→ volume
	↔ mm	↔ author	↔ title	↔ year	↔ 0000
		↔ editor			
		↔ translator			
		↔ sorttitle			
		↔ title			
nyt	presort	→ sortname	→ sortyear	→ sorttitle	→ volume
	↔ mm	↔ author	↔ year	↔ title	↔ 0000
		↔ editor			
		↔ translator			
		↔ sorttitle			
		↔ title			
nyvt	presort	→ sortname	→ sortyear	→ volume	→ sorttitle
	↔ mm	↔ author	↔ year	↔ 0000	↔ title
		↔ editor			
		↔ translator			
		↔ sorttitle			
		↔ title			
all	presort	→ sortkey			
	↔ mm				

C.2 Alphabetic Schemes 2

The following table shows the alphabetic sorting schemes for alphabetic styles defined by default. Please refer to § 3.5 for explanation.

Option	Sorting scheme					
anyt	presort	→ labelalpha	→ sortname	→ sortyear	→ sorttitle	→ volume
	↪ mm		↪ author	↪ year	↪ title	↪ 0000
			↪ editor			
			↪ translator			
			↪ sorttitle			
			↪ title			
anyvt	presort	→ labelalpha	→ sortname	→ sortyear	→ volume	→ sorttitle
	↪ mm		↪ author	↪ year	↪ 0000	↪ title
			↪ editor			
			↪ translator			
			↪ sorttitle			
			↪ title			
all	presort	→ labelalpha	→ sortkey			
	↪ mm					

C.3 Chronological Schemes

The following table shows the chronological sorting schemes defined by default. Please refer to § 3.5 for explanation.

Option	Sorting scheme				
ynt	presort	→ sortyear	→ sortname	→ sorttitle	
	↪ mm	↪ year	↪ author	↪ title	
		↪ 9999	↪ editor		
			↪ translator		
			↪ sorttitle		
			↪ title		
yndt	presort	→ sortyear (desc.)	→ sortname	→ sorttitle	
	↪ mm	↪ year (desc.)	↪ author	↪ title	
		↪ 9999	↪ editor		
			↪ translator		
			↪ sorttitle		
			↪ title		
all	presort	→ sortkey			
	↪ mm				

D 选项集合

The following table provides an overview of the scope (global/per-type/per-entry) of various package options.

Option	Scope			
	Load-time	Global	Per-type	Per-entry
abbreviate	•	•	–	–
alldates	•	•	–	–
arxiv	•	•	–	–

Option	Scope			
	Load-time	Global	Per-type	Per-entry
autocite	•	•	–	–
autopunct	•	•	–	–
autolang	•	•	–	–
backend	•	–	–	–
backref	•	•	–	–
backrefsetstyle	•	•	–	–
backrefstyle	•	•	–	–
bibencoding	•	•	–	–
bibstyle	•	–	–	–
bibwarn	•	•	–	–
block	•	•	–	–
citecounter	•	•	–	–
citereset	•	•	–	–
citestyle	•	–	–	–
citetracker	•	•	–	–
clearlang	•	•	–	–
datamodel	•	–	–	–
dataonly	–	–	•	•
date	•	•	–	–
dateabbrev	•	•	–	–
datezeros	•	•	–	–
defernumbers	•	•	–	–
doi	•	•	–	–
eprint	•	•	–	–
eventdate	•	•	–	–
firstinits	•	•	–	–
hyperref	•	•	–	–
ibidtracker	•	•	–	–
idemtracker	•	•	–	–
indexing	•	•	•	•
isbn	•	•	–	–
labelalpha	•	•	•	–
labelnamefield	–	–	–	•
labelnumber	•	•	•	–
labeltitle	•	•	•	–
labeltitlefield	–	–	–	•
labeltitleyear	•	•	•	–
labeldate	•	•	•	–
language	•	•	–	–
loadfiles	•	•	–	–
loccittracker	•	•	–	–
maxalphanames	•	•	•	•
maxbibnames	•	•	•	•
maxcitenames	•	•	•	•
maxitems	•	•	•	•
maxnames	•	•	•	•
maxparents	•	•	–	–
mcite	•	–	–	–
minalphanames	•	•	•	•
minbibnames	•	•	•	•
mincitenames	•	•	•	•
mincrossrefs	•	•	–	–
minitems	•	•	•	•
minnames	•	•	•	•
natbib	•	–	–	–
notetype	•	•	–	–

Option	Scope			
	Load-time	Global	Per-type	Per-entry
opcittracker	•	•	–	–
openbib	•	•	–	–
origdate	•	•	–	–
pagetracker	•	•	–	–
parenttracker	•	•	–	–
punctfont	•	•	–	–
refsection	•	•	–	–
refsegment	•	•	–	–
safeinputenc	•	•	–	–
singletitle	•	•	•	–
skipbib	–	–	•	•
skipbiblist	–	–	•	•
skiplab	–	–	•	•
skiplos	–	–	•	•
sortcase	•	•	–	–
sortcites	•	•	–	–
sortfirstinits	•	•	–	–
sorting	•	•	–	–
sortlocale	•	•	–	–
sortlos	•	•	–	–
sortupper	•	•	–	–
style	•	–	–	–
terseinits	•	•	–	–
texencoding	•	•	–	–
uniquelist	•	•	•	•
uniquename	•	•	•	•
urldate	•	•	–	–
url	•	•	–	–
useprefix	•	•	•	•
use<name>	•	•	•	•

E 更新历史

This revision history is a list of changes relevant to users of this package. Changes of a more technical nature which do not affect the user interface or the behavior of the package are not included in the list. If an entry in the revision history states that a feature has been *improved* or *extended*, this indicates a modification which either does not affect the syntax and behavior of the package or is syntactically backwards compatible (such as the addition of an optional argument to an existing command). Entries stating that a feature has been *modified*, *renamed*, or *removed* demand attention. They indicate a modification which may require changes to existing styles or documents in some, hopefully rare, cases. The numbers on the right indicate the relevant section of this manual.

3.0 2015-04-20

Improved Danish (Jonas Nyrup) and Spanish (Iudenticus) translations

`labelname` and `labeltitle` are now resolved by Biblatex instead of Biber for more flexibility and future extensibility

New <code>\entryclone</code> sourcemap verb for cloning entries during sourcemapping		
4.5.2		
New <code>\pernottype</code> negated per-type sourcemap verb	4.5.2	
New range calculation command <code>\frangelen</code>	4.6.4	
New bibliography context functionality	3.6.11	
Name lists in the data model now automatically create internals for		
<code>\ifuse<name></code> tests and booleans	3.1.3.1 and 4.6.2	
2.9a 2014-06-25		
<code>resetnumbers</code> now allows passing a number to reset to	3.6.2	
2.9 2014-02-25		
Generalised shorthands facility	3.6.4	Biber only
Sorting locales can now be defined as part of a sorting scheme	4.5.5	Biber only
Added <code>sortinithash</code>	4.2.4.1	Biber only
Added Slovene localisation (Tea Tušar and Bogdan Filipič)		
Added <code>\mkbibitalic</code>	4.10.4	
Recommend <code>begentry</code> and <code>finentry</code> bibliography macros	4.2.3	
2.8a 2013-11-25		
Split option <code>language=auto</code> into <code>language=autocite</code> and		
<code>language=autobib</code>	3.1.2.1	Biber only
2.8 2013-10-21		
New <code>langidopts</code>	2.2.3	Biber only
hyphenation field renamed to <code>langid</code>	2.2.3	
<code>polyglossia</code> support		
Renamed <code>babel</code> option to <code>autolang</code>	3.1.2.1	
Corrected Dutch localisation		
Added <code>datelabel=year</code> option	3.1.2.1	
Added <code>datelabelsource</code> field	4.2.4.1	
2.7a 2013-07-14		
Bugfix - respect <code>maxnames</code> and <code>uniquelist</code> in <code>\finalandsemicolon</code>		
Corrected French localisation		
2.7 2013-07-07		
Added field <code>eventtitleaddon</code> to default <code>datamodel</code> and styles	2.2.2	
Added <code>\ifentryinbib</code> , <code>\iffirstcitekey</code> and <code>\iflastcitekey</code>	4.6.2	

Added postpunct special field, documented multiprenote and
 multipostnote special fields 4.3.2

Added `\UseBibitemHook`, `\AtEveryMultiCite`, `\AtNextMultiCite`,
`\UseEveryCiteHook`, `\UseEveryCitekeyHook`,
`\UseEveryMultiCiteHook`, `\UseNextCiteHook`,
`\UseNextCitekeyHook`, `\UseNextMultiCiteHook`,
`\DeferNextCitekeyHook` 4.10.6

Fixed `\textcite` and related commands in the numeric and verbose styles 3.7.2

Added multicite variants of `\volcite` and related commands 3.7.6

Added `\finalandsemicolon` 3.9.2

Added citation delimiter `\textcitedelim` for `\textcite` and related commands
 to styles 4.10.1

Updated Russian localization (Oleg Domanov)

Fixed Brazilian and Finnish localization

2.6 2013-04-30

Added `\printunit` 4.7.1

Added field `clonesourcekey` 4.2.4.1 Biber only

New options for `\DeclareLabelalphaTemplate` 4.5.4 Biber only

Added `\DeclareLabeldate` and retired `\DeclareLabelyear` 4.5.9 Biber only

Added `nodate` localization string 4.9.2.14

Added `\rangelen` 4.6.4

Added starred variants of `\citeauthor` and `\Citeauthor` 3.7.5

Restored original `url` format. Added `urlfrom` localization key 4.9.2.15

Added `\AtNextBibliography` 4.10.6

Fixed related entry processing to allow nested and cyclic related entries

Added Croatian localization (Ivo Pletikosić)

Added Polish localization (Anastasia Kandulina, Yuriy Chernyshov)

Fixed Catalan localization

Added smart “of” for titles to Catalan and French localization

Misc bug fixes

2.5 2013-01-10

Made `url` work as a localization string, defaulting to previously hard-coded value
 ‘URL’.

Changed some Biber option names to cohere with Biber 1.5.

New `sourcemap` step for conditionally removing entire entries 4.5.2 Biber only

Updated Catalan localization (Sebastià Vila-Marta)

2.4 2012-11-28

Added `relatedoptions` field 4.5.1 Biber only

Added `\DeclareStyleSourcemap` 4.5.2 Biber only

Renamed `\DeclareDefaultSourcemap` to `\DeclareDriverSourcemap`
4.5.2 Biber only

Documented `\DeclareFieldInputHandler`,
`\DeclareListInputHandler` and `\DeclareNameInputHandler`.

Added Czech localization (Michal Hoftich)

Updated Catalan localization (Sebastià Vila-Marta)

2.3 2012-11-01

Better detection of situations which require a Biber or \TeX re-run

New append mode for `\DeclareSourcemap` so that fields can be combined 4.5.2 Biber only

Extended auxiliary indexing macros

Added support for plural localization strings with `relatedtype` 4.5.1 Biber only

Added `\csfield` and `\usefield` 4.6.1

Added starred variant of `\usebibmacro` 4.6.4

Added `\ifbibmacroundef`, `\iffieldformatundef`,
`\iflistformatundef` and `\ifnameformatundef` 4.6.4

Added Catalan localization (Sebastià Vila-Marta)

Misc bug fixes

2.2 2012-08-17

Misc bug fixes

Added `\revsdnamepunct` 3.9.1

Added `\ifterseinits` 4.6.2

2.1 2012-08-01

Misc bug fixes

Updated Norwegian localization (Håkon Malmedal)

Increased data model auto-loading possibilities 4.5.3 Biber only

2.0 2012-07-01

Misc bug fixes

Generalised `singletitle` test a little 4.6.2 Biber only

Added new special field `extratitleyear` 4.2.4 Biber only

Customisable data model	4.5.3	Biber only
Added <code>\DeclareDefaultSourcemap</code>	4.5.2	Biber only
Added <code>labeltitle</code> option	3.1.2.3	Biber only
Added new special field <code>extratitle</code>	4.2.4	Biber only
Made special field <code>labeltitle</code> customisable	4.2.4	Biber only
Removed field <code>reprinttitle</code>	3.4	Biber only
Added related entry feature	3.4	Biber only
Added <code>\DeclareNoinit</code>	4.5.7	Biber only
Added <code>\DeclareNosort</code>	4.5.8	Biber only
Added sorting option for <code>\printbibliography</code> and <code>\printshorthands</code> 3.6.2		Biber only
Added <code>ids</code> field for <code>citekey</code> aliasing	2.2	Biber only
Added <code>sortfirstinits</code> option	3.1.2.3	Biber only
Added data stream modification feature	4.5.2	Biber only
Added customisable labels feature	4.5.4	Biber only
Added <code>\citeyear*</code> and <code>\citedate*</code>	3.7.5	