

PAUL STANLEY

BIBLATEX

AN EASIER READ

2017

Copyright © Paul Stanley 2017. The moral right of the author is asserted.

This handbook is released under a CC Attribution-ShareAlike license CC BY-SA: <http://creativecommons.org/licenses/by-sa/3.0/>. In summary, you are free to

- Share — copy and redistribute the material in any medium or format
- Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

- Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

Typeset using L^AT_EX in the tufte-book class, using Palatino, Helvetica, and Bera Mono fonts.

Contents

	<i>Preface</i>	5
1	<i>What is Biblatex?</i>	7
2	<i>The Database File</i>	15
3	<i>Customisation: A Short Tour</i>	33
4	<i>Styles: The Menu</i>	37
5	<i>Citation Commands</i>	51
6	<i>The Bibliography</i>	61
7	<i>Multiple Bibliographies</i>	67
8	<i>Sorting</i>	77
9	<i>Languages</i>	81
10	<i>Recipes</i>	87

11	<i>Tools and Editors</i>	99
12	<i>When Things Go Wrong</i>	103
	<i>Quick Start Guide</i>	109
	<i>Aide Memoire or Cheat Sheet</i>	113
	<i>Examples</i>	115
	<i>Index</i>	130

Preface

The Biblatex package is a tour de force by its originator (Philip Lehmann) and its current maintainer(s) (Phil Kime — who is also responsible for Biber — assisted by others). It is powerful. But with power comes complexity. The manual is a mine of information, but sometimes rather overwhelming.

My aim here has been to write something that is a bit more than a mere introduction, but certainly not a systematic manual. That, after all, Biblatex already has. It is supposed to be, above all, practical: focussed on explaining not how Biblatex works, or exploring all its possible options and wrinkles, but trying to show how ordinary users can use it to accomplish reasonably ‘standard’ tasks. It is not intended to replace the manual; indeed, I have assumed that the reader will have that to hand. It is not intended as an advanced book for bibliography-style writers. It is aimed at the ordinary user, who is looking for practical advice about everyday issues. I hope it will be useful.

It is a pleasure to acknowledge the help of all those, particularly those who participate in `TEX-StackExchange`, who have directly or indirectly helped in so many ways. I am also grateful to all those who have commented on this guide’s Github project page. I welcome all comments and suggestions.

At the time of writing, the version of Biblatex on my system is 3.7.

PMS

London, February 13, 2019

pstanley@essexcourt.com

1

What is Biblalex?

This chapter has two different sections, intended for different audiences. If you are new to automated bibliography tools in \LaTeX , then you should read the first section. If you already have experience of using automated bibliography tools in \LaTeX , such as \BIBTeX and \Natbib , then you can probably turn straight to the second section. If you prefer to jump in and start getting your hands dirty without much explanation, there is a quick start section at the back of this guide, along with an aide memoire to the frequently used commands and options.

► p 7

► p 11

► p 109

► p 113

For the true neophyte

The basic idea

Academic writing usually cites sources. The exact form of these citations varies: different disciplines have different practices, and different publishers and journals too. But the underlying idea is similar: there are references in the text (or footnotes) that enable the reader to identify a source for a particular statement or quotation. This may be a number in a list of references (like [1]), or a combination of author and year (like Author 2014), or even a footnote which identifies the source.¹ Alongside this there is a bibliography: a list of all the sources which have been cited, which will nearly always enable the reader to look up the source in a library, and sometimes provide (in itself) useful information about the source, such as its date, or where and how it was published.

It is, of course, possible to produce all these indications entirely ‘by hand’: typing out references, bibliographies and so forth. But there are disadvantages to doing that:

- It is time-consuming. If you write lots of papers, you will probably find yourself citing the same sources again and again. It’s tedious to have to keep looking up and typing out the same data. You

¹ Like: Author, ‘Paper Title’ in *Learned Journal*, Vol. 10 (2014), p. 111.

have to spend time doing things like organising your bibliography into the proper order.

- It is error-prone. Every time you type data up, you risk making a mistake: either a mistake of substance (typing ‘Stanely’ for ‘Stanley’), or in the way the data is formatted.
- It can be frustrating. Each citation system has its own picky requirements. One wants the titles of articles in double quotes, “Like this”. Another wants them in single quotes, ‘Like this’. Another wants them in italics *Like this*. There are specialists who revel in such details: but for the average person they are just a nuisance, and it is nice to have a system which takes care of these details for you.
- The details sometimes change as you write. For instance, if you use a system where references are numbered, adding a reference may change all the numbering. Or if you follow a system which uses abbreviations such as ‘ibid’, ‘op. cit.’ or references back to earlier notes, adding a single footnote can throw everything out. In such systems it’s not possible to finalise references until the document is complete: and every change can easily introduce errors.

The idea of a citation system is to *separate content from presentation*. You record the important details about a source in a database file: the author(s), title, date, publisher and so forth. As you write, you use a short reference to indicate the work you want to cite at that point in your paper. And then you let the computer take care of extracting the information from your database, then inserting and formatting it into your paper. And in fact, in many fields, you can find existing databases which you can borrow from which will contain data you need.

This is what Biblatex does. You maintain a database which contains the works you may want to cite. The database can be large: it needn’t include only the works that you want to cite in a particular paper — it can be as large as you like. It can be in any order. All that matters is that it contains the information that is needed to ‘construct’ the citations.

As you write your paper, you insert commands to tell Biblatex that you want to cite particular sources drawn from that database.

Then, when you are done, as part of the compilation of your \LaTeX source, you get Biblatex to extract the relevant information, format it, and construct the citations and bibliography in proper format.

In practice

In practice, getting all this to happen involves a number of different tools and programs, which all have to work together.

You are going to start with two files:² a database file and your L^AT_EX source file.

The database file conventionally has the suffix `.bib`, and consists of a set of records containing information about the sources you may wish to cite. The format of such a file is described in chapter 2. It is a plain text file, which you can either produce by hand (using a text editor, such as the one you use to write L^AT_EX source code) or can be maintained by one of a number of available ‘helper’ programs.

The L^AT_EX source file conventionally has the suffix `.tex` and contains all your text, marked up in L^AT_EX format. It also contains:

- A line that loads Bibl_{at}ex and tells it what ‘style’ to use for citations and bibliography.
- One or more lines which tell Bibl_{at}ex what file(s) to use as the database from which information is to be extracted.
- One or more commands which tell Bibl_{at}ex that you want to `\cite` a particular source at that point in your text.
- (Usually) one or more commands that tell Bibl_{at}ex to print a bibliography at that point in the document.

² At least two, since your L^AT_EX source could `\include` or `\input` more than one file, and your database may consist of more than one file too.

► Chapter 11

```
\usepackage[style=...]{biblatex}
```

```
\addbibresource{file.bib}
```

```
\cite{...}
```

```
\printbibliography
```

The next task is to get the various tools to cooperate together to get to the end result that you need. In general that requires at least the following things to happen:

- Run L^AT_EX first on the source file. At this point no citations are produced. Instead Bibl_{at}ex *records* a list of the sources that you have cited.
- Run another program which reads the list that has been produced, analyses the database, and extracts the relevant information into a format that Bibl_{at}ex can work with. There is a choice of programs that you might use for this, but in this handbook we will be assuming that you will use a program called Biber.
- Run L^AT_EX again, to ‘read in’ the data that has now been put in digestible form, and produce the citations and bibliography.
- (Sometimes), run L^AT_EX yet *again* to finalise cross-references.

This ‘pattern’ (L^AT_EX, Biber, L^AT_EX, [L^AT_EX]) is repeated whenever you need to compile your document.³

³ Not quite true: it is often unnecessary to re-run Biber if no new citations have been added. But it is never wrong to do so.

A quick demonstration

Why not try a quick example? I assume for these purposes that you have a fully-equipped and functional \TeX system, and have installed the pre-requisites for Biblatex. This will be true on any of the standard modern \TeX installations; but it's not a bad idea to update to at least the most recent versions of Biblatex and Biber, since both are under active development.

First, let's set up our 'database'. It's not going to be much: just a single book.

Open a text editor, and produce a new file `handbook.bib`:

```
@book{nussbaum:95,
  author = "Nussbaum, Martha C.",
  title = "Poetic Justice",
  subtitle = "The Literary Imagination and Public Life",
  publisher = "Beacon Press",
  location = "Boston",
  date = "1995",
}
```

► Much more about how to write .bib files is explained in chapter 2.

Now open a text editor or your \LaTeX editor, and create a small test file (I'll call mine `test.tex`):

```
\documentclass{article}
\usepackage{csquotes}
\usepackage[style=numeric]{biblatex}
\addbibresource{handbook.bib}
\begin{document}
```

```
As Nussbaum comments \cite[17]{nussbaum:95}:
\enquote{The utilitarian picture of human beings and
of rationality is familiar enough in theory}.
```

► Much more about `\cite` commands is explained in chapter 5.

```
\printbibliography
\end{document}
```

Now, run \LaTeX on the file, once. If you look at the resulting type-set document, it should look something like figure 1.1.

As Nussbaum comments [**nussbaum:95**]: “The utilitarian picture of human beings and of rationality is familiar enough in theory.”

Figure 1.1: Before running Biber

The reference '`[sebum:95]`' appears in square brackets because although \LaTeX can 'see' that there is going to be a citation, it doesn't

yet have the data that will enable it to construct that citation, since Biber has not been run.

Now, from a terminal window opened in the same directory, run

```
biber test
```

Hopefully, you will see a number of messages marked as ‘INFO’, ending with INFO - Output to test.bbl.

Now run \LaTeX again on the source file test.tex. And if all is going well, you should now see output like figure 1.2.

As Nussbaum comments [1, p. 17]: “The utilitarian picture of human beings and of rationality is familiar enough in theory.”

References

- [1] Martha C. Nussbaum. *Poetic Justice. The Literary Imagination and Public Life*. Boston: Beacon Press, 1995.

Figure 1.2: After running Biber and \LaTeX

As you can see, the citation data has been pulled into the bibliography. If the output remains unchanged (it looks like figure 1.1) that is because Biber has not successfully run. That is *probably* because there is some error in your .bib file, so go back and check that. Any time you see this, you should ask yourself these questions: (1) have I told Biblalex where to find the bibliography file (using `\addbibresource{}`)? (2) Have I run \LaTeX then Biber then \LaTeX again? (3) Did Biber report any errors? (If you like, you can check the log file at `<jobname>.blg`) (4) Is my citation *key* correct (e.g., you haven’t typed nusbaum for nussbaum in the \LaTeX source)? (5) Does the .bib file include that citation? (6) Is the syntax of the citation correct?

Debugging

1. `\addbibresource`?
 2. Run \LaTeX Biber, \LaTeX ?
 3. Biber was error-free?
 4. Cite key is correct?
 5. .bib file contains source?
 6. Entry in .bib file valid?
- See chapter 12.

What next

There is still quite a bit to learn, of course. To some extent it’s up to you how you go from here, but the following chapters take what I think is a reasonably logical order.

For the more experienced

From the user’s perspective, the basic pattern followed by Biblalex is similar to the one you will be accustomed to if you use \BibTeX to produce .bst files, or citation packages such as Natbib:

- Your bibliographical data is still stored in .bib files, which have largely the same format as \BibTeX files. (But Biblalex recognises additional entry types and fields.)

- You still generate citation and bibliography data by running \LaTeX then an external program (either \BibTeX itself or a more powerful replacement, Biber, and then \LaTeX again (sometimes twice).

There are a few, largely cosmetic, changes to the commands that you need to include in your document to ‘set up’ the bibliography. (See page 13.)

Differences

So, what are the main differences?⁴

- There is a big difference in the essential way in which Biblatex and traditional \BibTeX -based systems work. In particular, in Biblatex the external program is used only to prepare data: the formatting and output of that data is largely handled using \LaTeX code, whereas in \BibTeX , most of the formatting is done by \BibTeX , producing a bibliography which is more-or-less ready to be typeset as it is.
- The practical consequence of that is that Biblatex can do things that traditional \BibTeX cannot: in particular it can respond *dynamically* to context in a way that traditional \BibTeX cannot, or cannot easily.
- This means that Biblatex can be used for citation systems (such as traditional humanities-style systems) for which \BibTeX alone is unsuited. It is a more powerful and flexible system.
- In addition, the Biber program – which is recommended to replace \BibTeX in sorting and handling the bibliographical data itself – is more flexible and powerful than \BibTeX is, and doesn’t suffer from some of the \BibTeX ’s historical limitations.

You may well be asking: should I use Biblatex? Most users probably fall into one of three groups.

Some people *need* to stick with traditional \BibTeX . For instance, if you are submitting work to a journal which has a \BibTeX style which it requires you to use, then you should *not* switch to Biblatex, at least for that paper. Many journals are in just that position, and very few (if any) have adopted Biblatex. And if you frequently or sometimes need to use \BibTeX , you should try to keep your `.bib` files compatible with it (for instance, continue to use `year`, `journal` and `address` rather than `date`, `journaltitle` and `location` fields).

Some people *don’t need* to switch. If you have existing \BibTeX styles which work for you, and do everything you need, then there is no reason to change. You can. But you don’t need to. The existing programs have the advantage of stability in such cases. On the other hand, switching will accustom you to Biblatex and put its more powerful features at your disposal. It’s really a matter of taste. Those who

⁴See also <http://tex.stackexchange.com/questions/25701/bibtex-vs-biber-and-biblatex-vs-natbib>

don't have an existing heavy investment in $\text{BIB}\text{T}\text{E}\text{X}$ should probably prefer Biblatex. But don't make such a change a week before you are due to submit your doctoral thesis.

Some people *need* Biblatex. This is probably true if you work in the humanities and need sophisticated and complex styles such as traditional footnote-based citations systems, if you want to cite non-standard sources, if you need to handle large volumes of non-ASCII characters or have complex sorting requirements, if you need to work in multiple languages.

The required changes

The main differences between using Biblatex and 'standard' $\text{BIB}\text{T}\text{E}\text{X}$ are as follows:

- You need to load Biblatex as a package using `\usepackage` (generally with various options). It's generally wise to load `csquotes` as well.
- Instead of `\bibliographystyle`, you specify the style to be used as an option when loading Biblatex:

```
\usepackage[style=...]{biblatex}
```

- Instead of specifying the `.bib` file as an argument to the `\bibliography` command, you use the command `\addbibresource{}` to identify the file(s). You specify the file(s) to be used completely, including their `.bib` suffixes. So if your bibliography file is called `mybib.bib`, you have

```
\addbibresource{mybib.bib}
```

- Instead of `\bibliography`, you use `\printbibliography` at the point where the bibliography should be printed.

One important point to understand: existing $\text{BIB}\text{T}\text{E}\text{X}$ *styles* cannot be directly used by Biblatex.⁵ The 'standard' styles in Biblatex do not precisely correspond to the standard styles in $\text{BIB}\text{T}\text{E}\text{X}$.

⁵ There is a project to provide Biblatex styles that are identical to the standard $\text{BIB}\text{T}\text{E}\text{X}$ ones: the `biblatex-trad` package. Development seems to have been sporadic.

An example

Find an existing `.bib` file of your own (or, if you are pushed to find one, make use of a standard file such as `biblatex-examples.bib`, which will be installed with Biblatex).

Try the following sample document:

```

\documentclass{article}
\usepackage{csquotes}
\usepackage[backend=bibtex, style=numeric]{biblatex}
\addbibresource{biblatex-examples.bib}
\begin{document}
\nocite{*}
\printbibliography
\end{document}

```

Although not essential, csquotes should normally be loaded.

Style is specified when loading the package, not using \bibliographystyle.

Instead of identifying the .bib files in the \bibliography command, they are identified this way.

Instead of \bibliography, \printbibliography is used to print the reference list.

If you run \LaTeX , \BibTeX , and \LaTeX again, you should find that a numbered bibliography is produced. Now see if you can get it to work with Biber. Replace line 2 with

```
\usepackage[backend=biber, style=numeric]{biblatex}
```

and delete .aux, .bbl and .blg files. Now run \LaTeX , Biber and \LaTeX again. You should generate the same document. (You might also like to try the sample document suggested for complete neophytes, which is at page 10.) Although Biblatex can use \BibTeX to prepare its data, it's generally better to use Biber, which is a more modern program with many advantages.

The Natbib package

Many \LaTeX users use the Natbib package. This is a sort of half-way house between \BibTeX and Biblatex. It's somewhat more flexible than a pure \BibTeX solution, and has (in particular) a wider range of citation commands to deal with author/year citation systems. But it still uses \BibTeX under the hood, and it doesn't have Biblatex's flexibility.

Biblatex has a Natbib 'compatibility mode'. If you load Biblatex with the option natbib (or natbib=true), then it will let you use some Natbib-like citation commands, like \citet and \citep. However, the compatibility is really only skin-deep; it hardly⁶ extends beyond the citation commands, and the actual formatting of the citations (which will depend on the style you specify) will be determined by Biblatex. In general, it's probably a better idea to get used to the Biblatex citation commands.

⁶ It does modify the punctuation used to separate an author's name from the year to match the Natbib default.

2

The Database File

This chapter is intended as a basic introduction to database files. It deals only with commonly used types of source, leaving the more obscure corners to the manual. It starts from the ground up, for those who have no `BIBTEX` or Biblatex experience. Those who are familiar with `BIBTEX` already can probably skim the first part, but should read the later parts fairly carefully because Biblatex with Biber is slightly different and considerably more powerful than `BIBTEX`.

The chapter gives examples of some `.bib` files and output they will produce. Do bear in mind that the precise form of the output is *dependent on the style chosen*: the examples are intended to help illustrate the general principles, but different styles may select from and format the data in different ways. The rest of the book is all about how to control such output. This chapter is about the input, and the examples are only there to give colour.

From ground up

The Biber program, which prepares data for use by Biblatex, can read a number of different formats; but only one is (at the moment) really fully supported, and that is the format you should use. It is a format originally developed for `BIBTEX`. It is simple. It can be written by hand, or produced with the assistance of specialised software.

The `BIBTEX` format database file conventionally has the suffix `.bib`. It is a ‘plain text’ file which (with Biber) can use any unicode characters.

It consists of a number of ‘entries’, each of which relates to a particular bibliographical source. Each entry consists of an *entrytype specifier* which explains what sort of source it is: like book, or article; a *key* which uniquely identifies the source and which you will use for citations; and a set of *fields* which contain bibliographical data about the work. The basic structure of such a record is shown in figure 2.1.

The *entrytype* specifier (`@book`) in figure 2.1) says what type of

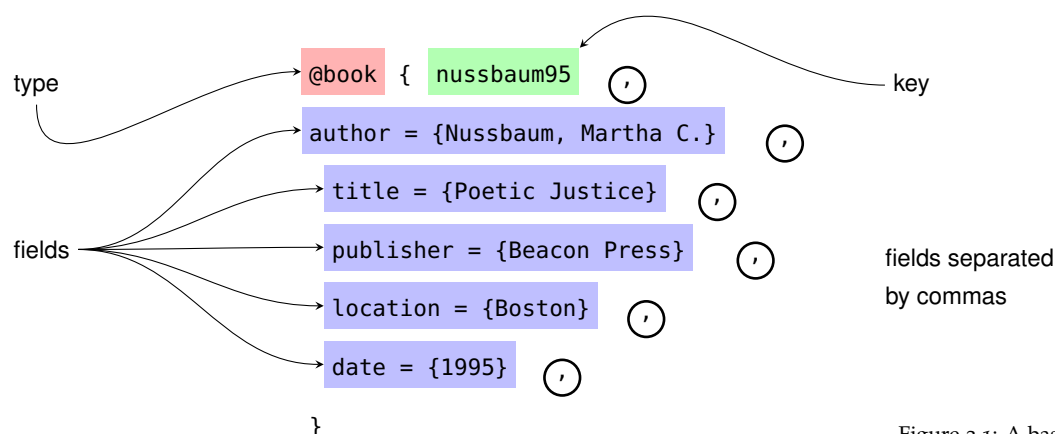


Figure 2.1: A basic source record

source this is. A large number of different types are supported, which we will look at in due course; indeed, in the end, the question is really what types are supported by the particular style you are using, since styles can define any type they like. But in this chapter we are going to look at just four types:

Book is used for an entry that consists of a complete physical book.

Article is used for journal articles.

Inbook is used for a self-contained chapter in a book.

Incollection is used for a self-contained paper in a collection of papers.

The entry type specifier is not case sensitive: @book, @Book and @BOOK all mean the same thing.

A source record, if you squint at it, takes the basic form

```
@entrytype{...}
```

The very first thing within the braces is the key. This does not have to take any particular form, but it does need to be *unique* to the particular source, and it should be something memorable enough and short enough to be practical for you, since you will be using it in documents you write.

Although a key can include quite a range of characters, there are some that you should avoid;¹ nor may a key include spaces. It's a good idea to be more-or-less consistent in the form your keys take, and to use common sense. For instance, nussbaum:1995 or nussbaum:poetic might be quite good keys, but nb could easily clash with others keys, nussbaum:1995:poetic-justice is arguably too long, and nbpoj95 may be hard to remember or work with. Use your common sense.

► See p 29 for the difference between inbook and incollection.

¹ " @ ' \ # { } ~ % _ & \$, ^

Immediately after the key, place a comma. The rest of the source record consists of a set of fields and values entered (usually)² in either the form

```
field = {value},
```

or, if you prefer, in the form

```
field = "value",
```

Fields must be separated by commas. It is permissible (though not required) to have a comma after the last one. I always make sure a comma is added to every field: it makes editing easier, because if you add another field you won't forget the comma.

The order in which you place the fields (apart from the key, which must come first) doesn't matter. You can use whitespace freely between fields and labels to keep things neat: as far as Biber is concerned

```
author = {Allen, Sidney},
title  = {Vox Latina},
```

is just the same as

```
author={Allen, Sidney},title={Vox Latina}
```

but for human consumption sensible spacing helps to keep things tidy.

Within a field you can use any unicode characters, and you can include L^AT_EX code too, if you need to (though it's sometimes a bad idea, especially in name fields, for reasons we will come to). Similarly, just as when you are writing L^AT_EX, you can break lines and end up with just a single space, which also helps you to arrange things neatly.

As figure 2.1 shows one can see that they are mostly self-explanatory. It's important to appreciate that they are intended as data sources: the raw material from which Biblatex will construct citations, not the citations themselves. So generally you should enter as much information as you have, because while Biblatex can ignore (or alter) information it has, it cannot invent information it doesn't have. For example, if entering a name you should, if you can, give the full name. Perhaps your currently-preferred citation system uses initials only. That doesn't matter, because Biblatex can extract the initials if it knows the full name; but it can never guess the full name from initials, and if you ever decided to use a system which needed the full name, you would be lost. Similarly, you shouldn't attempt any sort of formatting in these fields: they are just data, and if you want names in small capitals or titles in italics you deal with that by giving instructions when the .bib file is *used*, not within it.

² Occasionally neither braces nor quote marks are needed.

Field types

When you read the Biblatex manual, you will see that it distinguishes between five main types of field: name fields, list fields, verbatim fields, date fields and other fields, with various others for good measure. This is not actually a very helpful distinction from the user's point of view, and I'd suggest that you think of fields in this way:

► Manual § 2.2.1

- *Names* These *are* special, and worth thinking about separately. Typical name fields are `author` and `editor` fields.
- *Dates* Again, the *are* special, and worth considering specially. Typical name fields are `date` and `urldate`.
- *Other bibliographical fields*, like `title`, `journaltitle`, `url` or `isbn`. These are all fields that will contain material which might (depending on your style) find its way into your printed bibliography.
- *Meta-data* fields, like `keywords` and `options`, in which you provide information about your source which may be used to help format it, but is not expected to find its way directly into print.

Names

Names turn out to be rather complex things. Suppose I have the advantage of being called Quentin William Ffortescue von Rumpelstiltskin, Jr. I probably care about all the parts of this name; but bibliographical software is particularly interested in the last name which will be used for sorting (Rumpelstiltskin) and the first names which will, if necessary, be used for sorting (Quentin William Ffortescue), and from which the initials (Q. W. F.) will be constructed; but it may also need to know how to print my surname including its 'von' part, and to include the 'Jr.' after my name.

All this can be done. It can even (usually) be inferred. It *usually* doesn't matter whether a name is entered as John Smith or Smith, John. If in doubt, the following rules will keep you straight. (Not all of them is strictly necessary; but they are quite simple.)

1. Always enter initials with full stops. If it sees A. Author, or Author, A., Biblatex³ knows that A is an initial. If it sees A Smith it might think that 'A' is a (very short) name. Since this *can* matter, usually when you least expect it, always include the dot. This doesn't mean that Biblatex will necessarily *print* any dot when it outputs the bibliography. That is controlled by the bibliography style, so you can safely include the dots in your .bib file and lose them later.

³ Biber, actually.

2. Always put any *von* (or equivalent, such as *van* or *de la*) before the last name, as shown above.
3. For names *without Jr. or numbers* use either ‘ordinary’ name order or *⟨last name⟩, ⟨first name or initials⟩*. So *von Author, A.* or *A. von Author*. I highly recommend consistency in this. And in fact I strongly recommend that you use the Last, First format, which avoids mistakes with ...
4. If there is a ‘junior part’ (like Jr. or III), you *must* use the ‘backwards’ form:

von Rumpelstiltskin, Jr., Quentin William Ffortescue

5. If what looks like two words should actually be treated as one, then enclose them in braces. For instance, if a last name is double-barrelled but not hyphenated, it needs to go in braces. The hyphenated Homburg-Williams will be fine, but we don’t want Mr. J. Homburg Williams to find himself as J. H. Williams, so make it `{Homburg Williams}, J.`
6. Similarly some names are not made to be broken at all, notably institutions, which sometimes claim authorship or something close to it. If the University of Oxford ‘writes’ a report, Biblatex will be inclined to think of it as ‘of Oxford, U’. Avoid this by enclosing the whole name in (extra) braces:

author = {{University of Oxford}}

7. For accented characters, either use unicode or enclose the accented letters in braces: `{Victor, Paul {\’E}mile}`. Of these techniques, the use of unicode accented characters—which Biber can handle, unlike BibTeX—is much to be preferred. If you do have to use T_EX accents, enclose the character in question (but only that character) in braces.

Often academic works have multiple authors or editors. In that case, enter all the names separated by ‘and’ (*not* commas).

Ardman, A. **and** Baptiste, B. **and** Carruthers, C.

You can also put `and` others yourself. But be careful. How many authors’ names get printed is heavily style-dependent. Some styles only want one or two authors before printing ‘et al’; others may want four or five, or always print every name. It’s easy for Biblatex to truncate names if it has more than it needs, but impossible for it to guess who wrote a paper if it hasn’t been told, so as always, enter all the information you have available.

► See p 79 for advice on how to make sure such names are properly sorted.

► See p 87 for more information on how many names will be printed.

Dates

Enter dates in the form YYYY-MM-DD. So the 28th day in of February 2012 is 2012-02-28. In general, most Biblalex styles will work with less-than-complete dates, and of course in many cases you won't know the full date (for instance, the date when a book was published): so 2012 is a valid date, as is 2012-02 (February 2012), as is 2012-02-28. (Of course, that's not how dates will be *printed*, unless you want them to be: it's just how they are entered.)

Occasionally you need to enter a range of dates. In that case, use the a forward slash to separate the range: February to March 2012 is 2012-02/2012-03, and so on.

Data fields

Data fields other than name fields do not normally present such difficulties, but there are a few points to bear in mind.

Some such fields (for instance the publisher and location fields) are in fact *lists*: in such cases, if you need more than one entity mentioned, use *and* between them, as with names.

Data fields may be important for sorting. For example, although in most cases the primary sorting is done using the author or editor field, the title field may sometimes come into play. This can pose additional challenges, because it is possible, especially if the field includes some L^AT_EX command, to bamboozle the sorting algorithm. In general, enclose L^AT_EX commands (other than accented letters) in *two* sets of braces, and use unicode to deal with accented letters wherever possible. So, for instance, we would enter: As with names, accents and diacritics are probably best handled with unicode.

► See p 79 for more information about how to control the way titles are sorted.

```
title = {The {\LaTeX} Companion}
      or
title = {{\A} l'ombre des jeune filles en fleurs}
      or (better)
title = {Du côté de chez Swann}
```

Data fields may be modified. For instance, some style files capitalize only the first word of a title, so that an entry of

The German Law of Obligations

would become

The german law of obligations

Where you know that a particular letter must never be changed, you should 'protect' it with braces:

The {German}⁴ Law of Obligations

⁴ You will sometimes see this done as {G}erman, but the method given in the text is better, because it ensures proper kerning.

Otherwise it's best to enter titles, at least if you write in English, with 'significant words capitalised', since Biblatex can easily remove all but the first capital, but cannot be relied on to carry out more sophisticated changes.⁵ As noted above, some styles put titles into 'sentence case'—with only the first word capitalised. If you don't want that, the right thing to do is not to attempt to outfox the style by manually protecting every capital with braces, because that makes your database file useless if you ever do want such changes. The right thing to do is to choose a different style, or modify it, to preserve your capitalization.

⁵ In other words, if asked to capitalise the first letters of *The German law of obligations* it would likely end up with *The German Law Of Obligations*, which is not correct.

Metadata

There is a wide variety of metadata that you may, for various reasons, include in a file, including options to guide Biblatex in formatting citations, and keywords and options to help structure biographies. In general there is no particular difficulty with this sort of information. It is just plain text, still included in braces, and (sometimes) a comma-separated list.

Of these various fields, three stand out for special mention at this point.

Specifying hyphenation patterns Different languages are hyphenated in different ways. If you are writing a document in English, but cite a book whose title is in French, L^AT_EX will have trouble hyphenating it correctly. You can specify the language in the `langid` field. Then, *provided you set the package option* `autolang=hyphen`, the entry will be hyphenated using the appropriate language.

► Chapter 9

Keywords The `keywords` field enables you to provide one or more keywords (separated by commas). These can be useful if you want to separate out different kinds of source. For instance, if you wanted to produce different bibliographies by topic for an annotated bibliography, you could assign sources to particular topics using keywords.

► Chapter 7

Other meta-data fields include `options`, `sorttitle`, `labeltitle`, `indextitle` and `indexsorttitle`. These are all used for specialised purposes to enable you to over-ride defaults in the way that Biblatex would construct citations, sort the bibliography, or index an entry. Some of these are discussed as appropriate later in this book, where their use is explained.

► Chapter 8

Sub-types Some bibliography styles want to handle, say, an article in a newspaper differently from an article in a journal. For this purpose there is an `entrysubtype` field. This can also be useful when prepar-

ing subdivided bibliographies, where it can be used to select which works get printed in which section.

Common entry types

This section is going to take a look at four of the commonly used types of entry: books, articles, and the `incollection` and `inbook` types. This is not intended to be a complete formal description of every possibility. For that you can consult the documentation. It is supposed to give practical guidance, especially for neophytes. Most of what is said should seem like common sense for anyone who has ever written an academic bibliography.

► *Manual* § 2.1.

Books

The bare minimum for any book is some sort of title. *The Bible* is a book, timeless as it is and un-named as its Author customarily remains. Most books, however, have more information, and most bibliographical styles require more information to be provided, if it is available. Typically aim to record the following:

The author(s) name(s) in the author field. If there is an editor but no author leave the field blank.

The editor(s) name(s) in the editor field. If there is no editor, leave the field blank. If there is both an author and an editor, include the editor's name as well as the author's.

The title in the title field. Mostly this is easy, but some works exist to annoy us. Dr Edwin Poppie-Cocke has just completed 'Grammatical Solecisms: Dangling Participles and Misplaced Modifiers', which is volume 234 of his treatise on 'Elementary Stylistics'. What, exactly, is its title? For our purposes,⁶ we divide it into three:

- The maintitle, which is the title of the multi-volume treatise.
- The title, which is the particular title of the book.
- The subtitle.

We therefore enter:

```
@book{poppycocke:2013,
  author    = {Poppie-Cocke, Edwin},
  maintitle = {Elementary Stylistics},
  title     = {Grammatical Solecisms},
  subtitle  = {Dangling Participles and Misplaced Modifiers},
```

⁶ This is a simplification: there is also a `titleaddon` field that can be used to print something after the title in a different font. And `maintitle` can also have its own `mainsubtitle` and `maintitleaddon` too! Happily, these are rarely needed.

```

volume    = 234,
}

```

It is better to divide title and sub-title in this way because, although it would work if we didn't, some citation styles use the title when referring back to the work on second and subsequent citations, and the title alone, without subtitle, is usually sufficient for this purpose, and less of a mouthful.

Where you have a multiple volume work, you have two choices.

(1) You can have a single entry in your database for the work in question with all its volumes. This would normally be appropriate where you are dealing with one work, published at one time, which happened to be subdivided. In that case, enter the number of volumes in the volumes field – and remember to identify the particular volume you are referring to when you cite the work.⁷ (2) You may have separate entries for each volume, where each volume is effectively a separate work. That is normally appropriate where the volumes are published at different times. In that case, put the volume in the volume field. Some works have 'parts' instead of, or as well as, volumes: in that case put the part in the part field.

⁷ There is also a `mvbook` entry-type, which is specifically designed for multiple-volume works. It's probably better, though not compulsory, to use it for such works: see the *Biblatex Manual* § 2.1.1.1.

Some self-contained works are published as part of a series. If you want to record that information, put the series in the series field.

The edition in the edition field. This is usually omitted for the first edition if only one edition was published. If this is a number, just enter the cardinal number: {2}, not second or 2nd — the right suffix will be added by Biblatex. If it's something like 'revised' or 'corrected', type that.

Publication information In the date field. The year will usually suffice. The older BibTeX style was to enter the date in the year field. That will work fine, but it's probably better to use `date`,⁸ which is 'correct'. In some cases you will want to record two dates: the date of original publication, and the date of the particular edition you are referring to. In that case, put the date of the particular edition in the date field, and the date of original publication in the `origdate` field.

⁸ Unless you need your `.bib` file to be usable in BibTeX in which case stick with year.

The place of publication should be specified in the `location` field. You can also use a field called `address`, which is provided for backward compatibility purposes (it was the field used in the 'original' BibTeX.)

The publisher's name goes in the publisher field. This is a 'list field', so that you can have more than one publisher, in which case you enter them as

```

publisher = {Publisher A and Publisher B and Publisher C},

```

In some disciplines it is not customary to print the publisher's name, or the place of publication, and you may think that's pretty sensible since in most cases they are of no interest. Even so, unless you are very certain that you will never want this information, it's better to fill it in. It's always fairly straightforward to tell Biblalex not to use information that it has available — but impossible to tell it to create information you haven't provided. So, if you think of the future, it's best to include both.

THOSE FIELDS COVER THE BASIC BIBLIOGRAPHICAL DATA APPLICABLE TO MOST BOOKS but there are other pieces of information that you may sometimes need or want to record.

Short titles and shorthands If a title is being printed in a bibliography, it makes sense to print it in its entirety, even if it is very long. But some citation schemes (such as those that refer to works by author and title) make repeated use of titles, and if the title is very long such repeated references can become tiresome. It may make sense, in such cases, to abbreviate the title of the work on second and subsequent citation, so that—say—P. Kempees, *A Systematic Guide to the Case-Law of the European Court of Human Rights 1960–1994* can become 'Kempees, *Systematic Guide*'. In such cases, you can specify a shorttitle.

The *shorthand* is a related, but subtly different idea. In some works (for instance, commentaries or monographs devoted to the study of a small number of books) or there may be such frequent reference to a particular source that it makes sense to have an abbreviation for it. You can define a special citation form, called a 'shorthand', which will be used to cite the work in question. The package will also keep track of shorthands and you can print a list of them, similarly to a bibliography, using the command `\printshorthands` where you want the list to be produced. A shorthand is included in the shorthand field.

► See the *Manual* § 3.6.3.

Shorthands may be summarized in a list of shorthands. They need not, therefore, be self-explanatory; the reader will encounter them frequently, and (if they are not standard in the field) can be expected to look them up. They should be fairly rare — reserved for special cases. Short titles on the other hand will not appear in any separate list, and you should be careful to make sure that they will be immediately identifiable to any reader from the bibliography, and reserve them for cases where the title is long. In a specialist work on Wittgenstein it may be sensible to define *PI* as an abbreviation for *Philosophical Investigations*, but it wouldn't make sense to define that as a short title, or indeed to define any short title for general use. And in the example given above, 'Systematic Guide' makes a good

short title, but ‘SGCL60–94’ would be quite confusing. It is usually better not to define any short title or shorthand until you are sure that you need them.

Translators, adapters, revisers. It is usual to have an author, and common to have an editor as well. It’s not uncommon to have other people who should be credited: translators, commentators and the like. There are dedicated fields for translator, annotator or commentator, into which you can put the name(s). If you have some oddity, you can make use of up to two fields called `editora` and `editorb`. These are ‘generic’ fields, into which you can put names. If you do that, you also need to fill in the `editoratype` (or `editorbtype` ...) fields, with the ‘role’ of the person. Biblatex recognises as possible roles: ‘editor’, ‘compiler’, ‘founder’, ‘reviser’ and ‘collaborator’, and the enigmatic figures of ‘redactor’ and ‘continuator’ too. You can add others as well, though it’s not a totally straightforward task.

► See *Manual* §§ 2.3.6, 3.8 and 4.9.1.

The result, as the following example (whose output is shown in figure 2.2) shows, can in theory include a very large number of different roles, where that is needed.

```
@book{team,
  author      = {Author, Arnold},
  editor      = {Editor, Edwin},
  translator  = {Translator, Theodore},
  commentator = {Commentator, Cuthbert},
  editora     = {Redaktor, Richard},
  editoratype = {redactor},
  editorb     = {Collaborator, Christopher},
  editorbtype = {collaborator},
  title       = {Team Players},
  date        = {2013},
  publisher   = {Pubco},
  address     = {Oxbridge},
}
```

Author, Arnold (2013). *Team Players*. Ed. by Edwin Editor. Red. by Richard Redaktor. In collab. with Christopher Collaborator. Trans. by Theodore Translator. With a comment. by Cuthbert Commentator. Oxbridge: Pubco.

Figure 2.2: Bibliographical entry showing author, editor, commentator, translator, redactor and collaborator

Electronic publication Works which were, in the past, exclusively printed are now often published electronically, either as well as or instead of paper publication. The Biblatex package tries to reflect that.

One possibility is that the work is available on the internet, at a URL. In that case, you can specify a URL in the `url` field. Many bibliographic schemes require that the *date* on which that URL was last checked is also given, and that date should therefore be included in the `urldate` field.

A second possibility is that the work is available in a specialised electronic repository, such as arXiv or JSTOR. If that is the case, you can make use of the `eprint`, `eprinttype` and `eprintclass` fields to provide a reference for the work. Not every bibliographic style will use these details (and they can always be ‘turned off’).⁹

Finally, you can if you wish, provide a Digital Object Identifier DOI, which may provide a more permanent record of an electronic source than a URL, since DOIs do not change.¹⁰

ISBNs If you think it is ever likely to be used, you can include a book’s ISBN (International Standard Book Number) in the `isbn` field. As with DOIs, not every style will print ISBNs (and they can always be turned off).¹¹

Other information The Biblatex package allows quite a wide range of other information to be included in a `.bib` file, though not all of it will be used in all styles. For instance, it is possible to include information about the original language of a book, its original title, the title of a reprint and so forth. Since most of this information is of interest only in narrow fields of unusual cases, the reader is referred to the manual for details. Three fairly common fields are, however, worth noting:

- `pubstate` is used to provide information about the publication state of a work that has not yet been ‘properly’ published. Standard styles should recognise (in decreasing ratio of hope to expectation) `inpreparation`, `submitted`, `forthcoming`, `inpress` and `prepublished` as valid options here, and print appropriate indications.
- `note` and `addendum` may be used to provide necessary bibliographical information, in a free form, which would otherwise not have a ‘home’, such as ‘reprinted with corrections from the 1724 edition’. The difference is that `note` will usually get printed somewhere in the middle of an entry, while `addendum` gets printed at the end (though the precise position depends on the particular style). So

```
@book{generic,
  author   = {Author, Albert},
  ...
```

► See *Manual* § 3.11.7.

⁹ by setting the option `eprint = false` when loading Biblatex.

¹⁰ `http://www.doi.org`. Again this feature can be turned off by setting `doi = false`.

¹¹ by setting the option `isbn = false` when loading Biblatex.

```

note      = {With a note},
addendum = {And an addendum},
}

```

produces something like figure 2.3.

Author, Albert (1935). *A Little Thing Wot I Wrote*. With a note. Camford:
Dreaming Spires. And an addendum.

Figure 2.3: Note and addendum

- annotation may be used to provide a lengthy annotation, for instance for use in annotated bibliographies.

Articles

The article is the next basic form. Indeed, for most academic work it is probably the most common. It is entered into the database using the article entry type.

In general an article (as opposed to its reference) has only two significant parts: the author(s) and title. Those are formatted just as for books.¹²

```

@article{mueck,
  author = {Mueck, A. O. and
            Seeger, H. and
            Wallwiener, D.},
  title  = {Comparison of the Proliferative Effects
            of Estradiol and Conjugated Equine
            Estrogens on Human Breast Cancer Cells
            and Impact of Continuous Combined
            Progestogen Addition},
}

```

¹² Except that maintitle is not used. The subtitle field could, however, be used.

That, as figure 2.4 shows, gets us only part of the way: we have the *article's* title and authors, but we still need to provide the details of the journal in which it is published.

Mueck, A. O., H. Seeger, and D. Wallwiener (n.d.). "Comparison of the Proliferative Effects of Estradiol and Conjugated Equine Estrogens on Human Breast Cancer Cells and Impact of Continuous Combined Progestogen Addition". In: ().

Figure 2.4: Article with author and title, but reference missing.

The publication details are entered using fields for:

- `journaltitle` for the name of the journal.¹³ (There is also a `journalsubtitle` field, though it is very seldom required.)
- `series` for the journal's series (if any).
- `volume` for the journal volume (if any).
- `number` for the journal number (if any).
- `issue` for the journal's issue (if any) (such as 'Spring' – there are also `issuetitle` and `issuesubtitle` fields, which can be used where a particular issue has a special title which ought to be cited). Use `number` strictly for numerical subdivisions and `issue` for anything else. No matter that the journal you are dealing with call this its 'Christmas number': you will call it the `issue = {Christmas}`. And no matter the journal says this is volume 5, `issue 2`: you will call it `number = 2`.
- `pages` for the pages occupied by the article in question.
- `date` for the date of the publication.
- `year` available only for backward compatibility with `BIBTEX`. Unless you need that, use the `date` field.
- `month` for the journal's month, if that matters for citation. This should be either an integer (1 = January, and so forth) *or* a three letter code which should be entered *without* braces: `jan` not `{jan}`. It is only there for backward compatibility with `BIBTEX`. Unless you need that, you can just include the month as part of the `date`.

So we can complete our partial citation:

```
@article{mueck,
  ...
  journaltitle = {Climacteric},
  volume      = {6},
  pages       = {221--227},
  date        = {2003},
}
```

And producing the result along the lines shown in figure 2.5.

Mueck, A. O., H. Seeger, and D. Wallwiener (2003). "Comparison of the Proliferative Effects of Estradiol and Conjugated Equine Estrogens on Human Breast Cancer Cells and Impact of Continuous Combined Progestogen Addition". In: *Climacteric* 6, pp. 221–227.

¹³ For backwards compatibility reasons, `journal` will work too, but prefer `journaltitle` unless you need `BIBTEX` compatibility.

Figure 2.5: Article

Other information Information such as DOI, ISSN (the equivalent of ISBN for journals) and ANNOTATIONS is the same as for articles.

Collections and Parts of Books

Papers are often published in books — collected papers, *festschriften*, and collections devoted to a particular topic.

For such works, Biblatex provides two entry types: `inbook` and `incollection`. The boundaries between them are slightly hazy. In theory a book is a work which has, as a work, one or more ‘authors’ who take collective responsibility for it, and `inbook` is the entry-type corresponding to a discrete and self-contained part of that, whereas a collection is an assembly of disparate contributions which will have an editor or editors, but no author(s) as such, and `incollection` corresponds to a discrete part of that. In practice — as it lies on the shelf — a collection is a book. So the key difference is whether the ‘unifying feature’ of the work is its author (`inbook`) or its editors (`incollection`). But almost everything said about the book type above can be assumed to apply to the collection type, and everything said here about `inbook` can be assumed to apply to `incollection` as well.

In practical terms, for both `inbook` and `incollection` types you have two choices:

- You can include all the information in a single entry. In that case author and title are assumed to refer to the author of the discrete unit that is being cited, while `bookauthor`, `editor`, and `booktitle` refer to the larger work. You add either `pages` or `chapter` to indicate the part of the book occupied by the discrete unit.
- You can separately specify the information for the larger work (as book or collection, using all the fields given above). You then specify the author and title of the sub-unit, together with `pages` or `chapter`, and use the `crossref` field to link the individual entry to the larger work.

So, for instance, using the first method we might have:

```
@inbook{sedley:skulls,
  title      = {Skulls and Crossbones},
  author     = {Sedley, Stephen},
  bookauthor = {Sedley, Stephen},
  booktitle  = {Ashes and Sparks},
  booksubtitle = {Essays on Law and Justice},
  date       = {2011},
  publisher  = {Cambridge UP},
```

```

location    = {Cambridge},
pages       = {131--138},
}

```

Alternatively, you could set things up as follows:

```

@book{sedley:ashes,
  title      = {Ashes and Sparks},
  subtitle   = {Essays on Law and Justice},
  author     = {Sedley, Stephen},
  date       = {2011},
  publisher  = {Cambridge UP},
  location   = {Cambridge},
}
@inbook{sedley:skulls,
  title      = {Skulls and Crossbones},
  author     = {Sedley, Stephen},
  pages      = {131--138},
  crossref   = {sedley:ashes},
}

```

A citation of `\cite{sedley:skulls}` will produce exactly the same output in either case. My fairly strong advice is to prefer the second method, because it is more flexible – it enables citation of the whole book separately, and it enables you easily to add additional chapters or papers as units of their own without difficulty. It also means that, once a certain threshold is reached, the entire book will be added to the bibliography, which is often a convenience to readers. (By default, the ‘parent’ work is added to the bibliography if two ‘children’ are cited. You can change this number by giving a value to `mincrossrefs` in the options to Biblatex. So to set it to include the parent only when four ‘children’ are cited, you would include `mincrossrefs = 4` when loading Biblatex.)

A summary by type of literature

The preceding section of the chapter has surveyed the book, article, inbook and incollection types, and indirectly collection too, in some detail (though without exploring every tiny corner, for which the manual is invaluable). There are many other types – some supported by the standard styles, and some used by specialist styles (for instance, styles for legal citation or which include support for citing sound recordings). Rather than work relentlessly through them, table 2.1 suggests what type you might use to cite a number of different types of literature. Where the entry type is marked by an asterisk, that means that it is not supported in the standard styles, and you

should probably be looking for a specialised style that does support it.

literature	suggested entry type	
article (journal)	article	
article (in collection)	incollection or inbook	
article (in newspaper)	article	(some styles offer special formatting)
article (unpublished)	article	use pubstate
book	book	
letter (unpublished)	misc	
letter (published)	article	
manual	report	
online item	online	
thesis	thesis	set the type for the type of thesis

Table 2.1: Sources and entry types

Replacing text

One curse of bibliography generation is the abbreviation. Suppose you have a journal called the ‘New York University Legal Studies Quarterly’. Sometimes you will be told not to abbreviate it at all. Sometimes you will be told to abbreviate it to ‘NY University Legal Studies Q’; sometimes to ‘NYU Legal Studies Q’. And so forth. (Slightly similar problems can apply with publishers: is it ‘Oxford Univ. Press’ or ‘OUP’ or ‘Oxford University Press’ or ‘Oxford UP’?)

If you work in a field where this is not a problem—where standard abbreviations are absolutely set in stone—then you are in clover. In other fields, some element of flexibility is needed. There are two ways you can achieve this, and one way that you can make your life as difficult as possible.

If you want to make your life difficult, be sure to enter articles in a radically inconsistent way, using different forms of the journal name or publisher.

The two ways you can make your life easy are as follows.

First, you can use *strings* in the .bib file. To do this, define a string at the top of the file as follows

```
@string{NYULSQ = "NY University Legal Studies Q"}
```

and then, in the journaltitle field, simply put NYULSQ (without braces)

```
@article{...
  journaltitle = NYULSQ,
```

```
...
}
```

the string definition will be substituted for the abbreviation before processing. That means that if you want to change all the entries to read ‘NYU Legal Studies Q’, you just have to change one line (in the string definition).

The alternative is to be, at least, completely consistent. Then you can use the power of Biber to search for and replace strings in your file. For instance, suppose that throughout your file you had entered the above-mentioned (fictitious) journal as NYU Legal StudiesQ, but you are now writing a paper in which the proper abbreviation is ‘NYU Leg Studs Q’. The following lines, placed in the preamble to your file, will accomplish the necessary change:

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{           % for .bib files
    \map[overwrite=true]{           % (over-writing if need be)
      \step[fieldsource=journaltitle, % if the journaltitle field
        match={NYU Legal Studies Q}, % reads "NYU Legal Studies Q"
        replace={NYU Leg Studs Q}] % replace it with "NYU Leg Studs Q"
    }
  }
}
```

A sourcemap directive is a powerful thing. It tells Biber to make changes to the input file data *before* outputting them for L^AT_EX to use. In this case, it tells it to apply the changes to any .bib file.¹⁴

The changes consist of a filtering step where it examines the journaltitle field in any source and, if that field matches NYU Legal Studies Q, Biber will replace it with NYU Leg Studs Q. This does not actually change your .bib file; but it alters the data as it passes from that file into Biblatex for formatting.

As you can see, this trick will only work if you have been *consistent* in the way you have named journals and publishers. But so long as you are consistent, it’s a powerful technique.

► Manual § 4.5.2.

¹⁴ Don’t suppose that datatype = bibtex means that the BIB_TE_X program can do this. It’s a Biber technique only!

3

Customisation: A Short Tour

We are about to dive into the details of using Biblatex. As we go through them, we are going to be thinking a bit about customisation.

Biblatex is highly customisable. That, in the end, is its point. At one extreme, if you are setting out to write a bibliography style for general use you might be customising a great deal. But at that point it gets very complicated. If you are trying to do anything that difficult you will really want to understand the internals of Biblatex — which will involve reading the manual carefully, and also looking at some of the source code.¹ This guide is not trying to teach you how to do that. But there are lots of customisations that are well within the reach of the ‘ordinary’ user, and it makes sense to discuss them. This chapter provides a general introduction. Further details are given throughout the guide in the places where they make most sense, and there is a chapter that deals specifically with ‘recipes’ for various commonly-needed changes.

In order to keep things simple, I am going quite often to explain *how* to obtain certain commonly needed changes, without necessarily explaining exactly why it works—what internal mechanism is in play.

However, to avoid frustration, you may need to understand some of the common ways in which changes are made, so that you can include the right commands in the right way. The final section of this chapter explains the bare minimum that you need to know.

Finally, do bear in mind that, with Biblatex there are often several ways to achieve a particular result. You may find other suggestions elsewhere about how to get Biblatex to do something, and they may well be just as good (or better) than the ways I have suggested. Generally, I’ve tried to keep things as simple as possible.

► *Manual* § 4

¹ Especially `standard.def` and the `.bbx` and `.cbx` files for the various standard files.

► Chapter 10

The common methods

Using an option. Many ‘customisations’ can be achieved simply by setting one of Biblalex’s package options. This is done when you load Biblalex. For instance, to change the way the bibliography is sorted, you might set the sorting option to none, by loading

```
\usepackage[style=numeric,
             sorting=none]{biblalex}
```

Apart from the package itself, some commands also have options. For instance, there are a number of options that can be given to the `\printbibliography` command to affect how the bibliography is printed.

Redefining a command. It’s quite common for a change to be made by redefining a command. For instance, in many cases Biblalex uses a command to insert a piece of punctuation, so that if you want to change the punctuation, you can redefine the command to produce a different punctuation mark.

For instance, there’s a command called `\multicitedelim` which inserts the punctuation between multiple citations. Styles set it differently, but it’s usually a comma or a semicolon. Suppose, for some really odd reason we wanted to make it ‘AND’. To do this, you use `\renewcommand`.

```
\renewcommand{\multicitedelim}{\addspace AND\space}
```

Punctuation. At this point, one feature of bibliography-bashing is worth attention. If you do much of it, you will find yourself spending quite a bit of time fiddling with punctuation. It’s not easy, but Biblalex has some rather sophisticated ways of helping you.

When setting punctuation in Biblalex commands, it’s often best *not* to use punctuation marks directly: for although they will work, they don’t ‘play’ nicely with some of the clever tricks Biblalex uses to keep punctuation straight. Instead, when engaged in customisation, use the various commands that Biblalex offers.

These (see table 3.1) all take the form `\add...`. Where they often have the edge on simply using punctuation marks directly, is that they are context sensitive: they will not add marks if it is inappropriate (for instance, they won’t add a comma after a question-mark), and they will remove unnecessary white space before the mark. The best way to think of them is as ‘context-sensitive’ punctuation.

Bibliography strings. Apart from the bits of bibliographical informa-

► Manual § 3.1

► See chapter 3

► See p 91

<code>\adddot</code>	.	(abbreviation dot)
<code>\addperiod</code>	.	(regular period)
<code>\addcomma</code>	,	
<code>\addcolon</code>	:	
<code>\addsemicolon</code>	;	
<code>\addexclam</code>	!	
<code>\addquestion</code>	?	
<code>\addslash</code>	/	
<code>\addspace</code>		(regular space)
<code>\addnbspace</code>		(unbreakable space)

Table 3.1: The `\add...` commands

► For spacing commands see the *Manual* § 4.7.4

► Manual § 4.9.1

tion assembled from the database, all sorts of bits of text get printed in a bibliography: ‘ed.’, ‘vol.’, ‘pp.’ and so on. These are handled by Biblatex using bibliography strings. One quite common customisation is to change these. This is done using the command

```
\DefineBibliographyStrings{<language>}
  {<bibstring> = <definition>},
  <bibstring> = <definition> }
```

For instance, if we decided that instead of printing ‘edited by’ or ‘ed by’, Biblatex would produce ‘conjured by’ we could alter the byeditor string:

```
\DefineBibliographyStrings{english}% or your language
  { byeditor = {conjured by}, }
```

Field formats. Internally, when it prints some data from the .bib file, Biblatex passes the data through a filter that is defined for that field and entry-type. Although there are some extra complexities about fields that hold names or lists, the basic filter is set up using a command `\DeclareFieldFormat`. The basic structure of this is

► *Manual* § 4.4.2

```
\DeclareFieldFormat[<entrytype>]{<field>}{<format>}
```

The `<format>` should just be the definition of a command which accepts a single input (`#1`). At its simplest, it might simply print this:

```
\DeclareFieldFormat{title}{#1}
```

would just print the field, whereas

```
\DeclareFieldFormat{title}{***#1***}
```

would print the title surrounded by three asterisks, or

```
\DeclareFieldFormat{title}{\textsc{#1}}
```

would print the title in small capitals.

Because of the way that Biblatex works there are also formatting directives for lists and names. These are more complex; ordinary users are probably best to leave them alone; but they may be encountered from time to time in particular recipes.

```
\DeclareListFormat
\DeclareNameFormat
```

Hooks. There are a number of ‘hooks’ offered by Biblatex at which it is made relatively easy to insert user-defined commands which might do something useful. For instance `\AtEveryCitekey` allows one to execute some code whenever a source is about to be cited, at a time when its data is available but has not yet been handled; `\AtEveryBibitem` offers a similar facility in relation to the printing

► *Manual* § 4.10.6

of bibliography items. Particularly useful, sometimes, is the hook `\AtDataInput`, which is executed when data is first ‘read in’ to become available for citation. From time to time you may be encouraged to use these hooks for various purposes.

Source mapping. In fact, Biblatex can ‘get at’ the data even before it is read in—indeed, while it is being processed by Biber. The facilities providing for source mapping enable one to play various useful tricks with the `.bib` file. We’ve already seen an example of this on page 32.

► *Manual* § 4.5.2

Deeper and darker. All these techniques (options, redefinition of ‘hook’ commands, and the redefinition of bibliography strings) are fairly simple, and intended for regular use. Some customisations take us deeper: into the internal workings of Biblatex. These things include the redefinition of citation commands, the redefinition of things called bibmacros, and ultimately the rewriting of bibliography drivers.

Where to put customisations

This all depends on whether your customisations are ‘one off’ or intended to be used in all your documents.

- For one off customisations:
 - customisation by options should go in the options that you list when loading Biblatex.
 - most other customisations should go in the document preamble, after Biblatex has been loaded but before `\begin{document}`. In a *very few* cases it’s better to put them at some other point. Where that is so I specifically mention it.
- If you want a standard set of options to apply to all your documents, put them in a file called `biblatex.cfg`, which should be put in your *local* TEXMF tree at `/tex/latex/biblatex`. If there is such a file it will be loaded after Biblatex. It can be used to redefine commands and to set (most, but not all) bibliography options, using `\ExecuteBibliographyOptions{<options>}`. You cannot, however, set the bibliography style there: that at least you must do when you load Biblatex.
- If you get to the point that you are developing a comprehensive style, you will want to write `.bbx`, `.ltx` and `.cbx` files which can be loaded as a style. That, however, is beyond the scope of this document.

► *Manual* § 4.2

4

Styles: The Menu

There are many different citation and bibliography styles available for Biblatex, and half the battle is finding one which meets your needs as closely as possible. But before looking at them in detail, it's a good idea to try to identify them broadly.

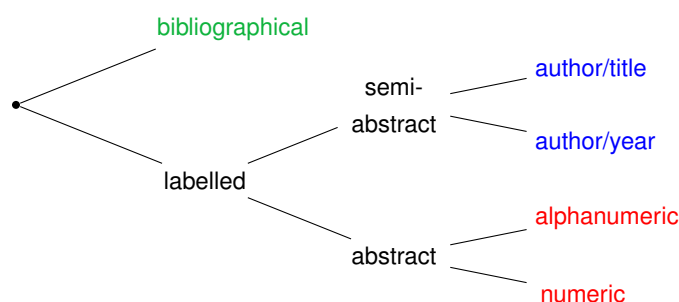


Figure 4.1: Citation: Family Tree

The first big division is between styles which rely on *labels* and those that place full bibliographical information directly into citations. In labelled styles the bibliography contains the essential information about a citation, and the text contains references which are not self-contained, but are intended to allow the reader to cross-refer to the bibliography. In contrast, in unlabelled styles, bibliographical information can be found in the citations themselves. It may be repeated in the bibliography. But the reader is not expected to need to look anything up in the bibliography to find information about the work cited. However, because this consumes so much space, it is common to find a system of cross-references and abbreviations (including, in extreme cases, a panoply of latin gadgets: *ibid.*, *op. cit.*, *loc. cit.*, *supra*, *infra*).

The next division is between *abstract* labelling systems and *partly meaningful* labelling systems. In an abstract labelling system, the label carries no bibliographical information at all. It might be simply a number. To follow it up, the reader *must* look it up in the bibliography. In a partly meaningful labelling system (typically an au-

thor/year system), the label does provide some information which is likely to mean something: it tells you who wrote the work, and when. But the information is still not comprehensive, and the reader who wanted to track down the specific source would still need to use the bibliography.

Table 4.1: Examples of citations in different families

	first citation	later citations
bibliographic	M. Nussbaum, <i>Poetic Justice</i> (Beacon Press: Boston, 1995)	Nussbaum, op. cit. supra n. x
		ibid.
author/title	Nussbaum, <i>Poetic Justice</i>	Nussbaum, <i>Poetic Justice</i>
author/year	(Nussbaum 1995)	(Nussbaum 1995)
alphabetic label	[Nus85]	[Nus85]
numeric	[1]	[1]

Of course, within each family there is considerable variation. Numerical systems may place numbers in brackets, or parentheses, or as superscript; they may sort and compress ranges, or leave them uncompressed; they may sort the bibliography alphabetically and then put references in, or they may present the bibliography in the order in which the works happen to appear in the text. Author/year systems may format their references in a variety of different ways. And full-bibliographical systems differ tremendously in the range of latin gadgets they regard as acceptable or necessary. But these differences, though important, are less pronounced than the fundamental differences between the types of citation.

Biblatex styles also fall into these three families, and there are a number of ‘standard’ styles in each. In principle, Biblatex distinguishes between the ‘citation style’ (which determines how citations appear in the text) and the ‘bibliography style’ (which determines how the list of references is printed; it is possible to specify each separately. In practice, however, there tends to be a connection between these things, and most users will not specify them separately, but will simply specify a style option when loading Biblatex which will load both citation style and bibliography style. That is what we assume here.

Another line of division within Biblatex styles is between generic styles and specific styles. Generic styles (like the standard Biblatex styles) reflect common citation practices but do not insist on conforming to any particular style guide. Specific styles aim to provide accurate implementations of particular detailed guides. In principle, there is much to be said for specific styles, which are more likely to be accurate reflections of citation practices in the field to which they relate.

[1] (1) ¹ [1, 5–8, 10] [1, 8, 10, 7, 6, 5]

(Author 1980) (Author: 1980) (Author, 1980)

supra n. θ , n θ above, (n θ)

`\usepackage[citestyle=...]{biblatex}`

`\usepackage[bibstyle=...]{biblatex}`

If you are looking for a Biblatex style the first question should be whether there is an existing style which aims to implement the particular citation style you are following. There are many such packages available on CTAN (a selection is given in table 4.2). However, you do need to be careful, because while some of the packages—for instance `biblatex-apa` and `biblatex-chicago`—are very stable and solidly maintained, others are less polished and mature. The only solution is to test.

style	package	version	date
AIP	<code>biblatex-phys</code>	1.0b	2016
APA	<code>biblatex-apa</code>	7.4	2017
APS	<code>biblatex-phys</code>	1.0b	2016
American Chemical Society	<code>biblatex-chem</code>	1.1s	2017
Angewandte Chemie	<code>biblatex-chem</code>	1.1s	2017
<i>Historische Zeitschrift</i>	<code>historische-zeitschrift</code>	1.1a	2014
Chicago	<code>biblatex-chicago</code>	1.0rc1	2016
IEEE	<code>biblatex-ieee</code>	1.2a	2017
MLA	<code>biblatex-mla</code>	1.9	2016
<i>Nature</i>	<code>biblatex-nature</code>	1.3b	2017
NEJM	<code>biblatex-nejm</code>	0.4	2011
Royal Society of Chemistry	<code>biblatex-chem</code>	1.1s	2017
<i>Science</i>	<code>biblatex-science</code>	1.1g	2016
OSCOLA	<code>oscola</code>	1.5	2017
Philosophy	<code>biblatex-philosophy</code>	1.9	2017
SBL	<code>biblatex-sbl</code>	0.8.1	2017

Table 4.2: Packages for Particular Styles

The second type of style is not tied to any concrete style guide or set of rules, but aims to reflect general academic practice and to provide a reasonable style which can be used or adapted. The standard styles that come with Biblatex are in this category. But there are also other styles, available on CTAN which are also like this. These styles often aim at providing a large amount of configurability and flexibility.

So how should you choose a style? The ideal situation is that the exact style you want already exists. There's a good chance that is the case. For instance, in the humanities, the Chicago styles (which offer both an author/year and a full bibliographical form) are very commonly used; and between the APA style (which is an author/year style), the MLA style (which is an author/title style) and the various styles for scientific journals, you may well find you are covered. If so, use that style (and consult its documentation carefully). Scientists are also well covered by a variety of different numeric styles. A variety have been illustrated with examples at the end of this document. The Biblatex documentation also includes many examples, together with the document that produced them.

See, in particular, the `biblatex-dw` styles.

► See page 115

If there isn't an exact style that is precisely what you want, you will have to find one that comes close to your needs, and either live with it or adapt it. Provided your requirements are not rigidly fixed,¹ you can probably find something which is satisfactory.

I'm not going to say much more of the precisely tailored styles: you'll know if those are what you want. The rest of this chapter surveys some of the standard (and a few of the other available 'general' styles) to help you understand what they provide. It's helpful to consider them family-by-family. But bear in mind that the main strength and purpose of Biblatex is not in its in-built styles, which are intended rather as starting points for adjustment than as a bibliographic *dernier cri*. Still, much of what is said about them is generally applicable.

Standard Numeric Styles

Numeric styles, though rather tedious to produce in the days of typewriters and hot metal type, are quite convenient for computers, and we will start with those. There's really room for variation in four areas:

- Whether the references are listed in order of (first) citation, or sorted into a logical order (usually, alphabetical by name of author).
- Exactly how the bibliographical entries are formatted internally (for instance whether journal articles are printed in italics, or put in quotation marks).
- The precise format of references in the text. Numbers, of course: but are they in brackets [1] or parentheses (1) or superscript¹? Are they perhaps in bold [1]. If there are multiple citations are they separated by commas [1, 2]; or perhaps by semi-colons [1]; [2]?
- How to handle ranges. If I cite items 1, 3, 4, 8, and 5, should I end up with the citation [1, 3, 4, 8, 5], or should it be sorted [1, 3, 4, 5, 8], or should it be sorted and compressed [1, 3–5, 8]?

How do the Biblatex standard numeric styles deal with these issues?

It's easiest to deal with the actual formatting of the bibliographical material first, because that doesn't change: all the numeric styles handle it in the same way. Figure 4.2 shows two very ordinary book and article references, formatted as the standard styles do.

If that doesn't look right for you, you will need either to try to identify an existing numeric bibliography style which is correct, or to adapt the standard in some way.

¹ If requirements *are* absolutely rigid, it sometimes turns out to be impossible to meet them precisely without adaptations that are not easy to make. Generally, if the adaptation requires re-ordering the information that is printed, it is likely to be fairly hard to do, whereas if it requires simple re-styling, it should be quite simple.

- [1] Frank Albert Cotton et al. *Advanced inorganic chemistry*. 6th ed. Chichester: Wiley, 1999.

[2] Trevor R. Reese. "Georgia in Anglo-Spanish Diplomacy, 1736–1739". In: *William and Mary Quarterly*. 3rd ser. 15 (1958), pp. 168–190.

Figure 4.2: Numeric bibliography style

SORTING IS THE NEXT THING to consider. In Biblatex that is dealt with by a package option which specifies the sorting method to be used. If you are used to `BIBTEX` this will be unfamiliar to you, because traditionally `BIBTEX` 'baked in' the sorting mechanism to the bibliography style, whereas in Biblatex it can easily be changed.

By default, the numeric styles in Biblatex sort the references into alphabetical order: looking first at the name of the author or editor, then at the title, and then at the year. If you want citations to be listed in *citation order*, you will need to specify the option

`sorting=none`

when you load Biblatex.

► See chapter 8 for much more information on sorting.

THAT TAKES US TO THE FORM AND ORDER OF REFERENCES, and at this point the standard styles do offer some options, particularly in terms of order.

1. If you want references *uncompressed* and *not reordered*, just load the style `numeric`. The package will then print the references you give in any citation in exactly the order you cited them in.
2. If you want references *sorted and re-ordered* but *not* compressed, then load the style `numeric`, but also add the option `sortcites=true`. (This, at least, you will surely nearly always want!)
3. If you want references *sorted and compressed*, then load the style `numeric-comp`

(There's also a style called `numeric-verb`. It's described at in the Biblatex documentation; but it's rather rarely useful.)

When it comes to changing the citation format, the most common requirements can be met as follows For *superscript citations*, you have three options:

1. Use the `\supercite` command instead of `\cite`: this will produce superscript citations.
2. Load Biblatex with the option `autocite=superscript` and then use the `\autocite` citation command instead of `\cite`.

	numeric	numeric-comp
<code>\cite{a}</code>	[1]	[1]
<code>\cite{a,b}</code>	[1, 2]	[1, 2]
<code>\cite{b,a}</code>	[2, 1]	[1, 2]
<code>\cite{a,b,c}</code>	[1, 2, 3]	[1–3]
<code>\cite{c,a,b}</code>	[3, 1, 2]	[1–3]

Table 4.3: Effect of compressing and sorting

`blah.\supercite{a} → blah.1`

3. Remap `\cite` to `\supercite` (or to `\autocite` with the superscript option set).

Changing the *punctuation between multiple citations* is easy. By default it's a comma. You change it by redefining `\multicitedelim`. So if you want a semicolon, you would put (in your preamble)

```
\renewcommand{\multicitedelim}{\addsemicolon\space}
```

Occasionally one sees styles where the label is in bold or some other funny font. If you have to do that, you need to provide field formats for `labelprefix`, `labelnumber` and `entrysetcount`. Each takes the same form (in the preamble of your document, assuming you wanted bold):

```
\DeclareFieldFormat{labelnumber}{\textbf{#1}}
```

Or whatever other format you want.

If you want to use *parentheses* instead of brackets around citation labels—so you get (1) instead of [1]—or if you want to dispense with these delimiters altogether, you need to do rather more work. This topic is covered in some detail below.

► See further, chapter 10

► p 93

Alphanumeric Labels

Alphanumeric labels look like [JW86] or [Jon95]. They are a half-way house between the elegant simplicity of numerical citations, and the additional information provided by author/year styles.

If you need or want to use such a system, load either the style `alphabetic` or `alphabetic-verb`. The only difference between the styles is how multiple citations are treated: with `alphabetic` you get them enclosed in a single set of brackets separated by commas whereas with `alphabetic-verb` you get each reference in its own set of brackets, separated by semicolons.

```
\cite{a, b}→[AAo1, BB02]
```

```
\cite{a, b}→[AAo1]; [BB02]
```

The printed bibliography is obviously essential to a style that uses alphanumeric labels, because the reader has to have a way of looking up what the labels mean. The bibliography will look something like figure 4.3

- | | |
|----------|--|
| [Cot+99] | Frank Albert Cotton et al. <i>Advanced inorganic chemistry</i> . 6th ed. Chichester: Wiley, 1999. |
| [Ree58] | Trevor R. Reese. "Georgia in Anglo-Spanish Diplomacy, 1736–1739". In: <i>William and Mary Quarterly</i> . 3rd ser. 15 (1958), pp. 168–190. |

Figure 4.3: A bibliography with alphanumeric labels

COMPRESSION OBVIOUSLY MAKES NO SENSE for labels (how would one do it?) but you can, if you like have them sorted by setting the option `sortcites=true`: they will be placed in the same order they appear in the bibliography. But you may be interested in how the label actually gets constructed.

If you don't like the labels that Biblatex produces in a particular case, you can override its decision in one of two ways.

First, if the shorthand field is set, then Biblatex will use it instead of the label. So, for instance, if we set

```
@book{book1,
  author = {Author, A.},
  date   = {1989},
  ...
  shorthand = {Masterpiece},
}
```

then `\cite{book1}` will produce [Masterpiece], not [AA89].

Alternatively, you can directly set the `label` field. In practical terms, that will produce the same result.

```
@book{book1,
  author = {Author, A.}
  ...
  label = {MyLabel}
}
```

and `\cite{book1}` produces [MyLabel89]. You can see the difference: whereas the shorthand field is used 'as is', the `label` field substitutes only the alphabetic part of the label: the year part is still added automatically.

A still more advanced change would be to alter altogether the way that labels are generated. This is an advanced customization, which is not covered in this document: consult the Biblatex documentation.

► *Manual* § 4.5.4

IN TERMS OF FORMATTING there's not much room for variation.

- You can change the punctuation that gets printed between multiple sources (by default, a comma in alphabetic and a semicolon in alphabetic-verb) by redefining `\multicitedelim`.
- You can alter the font in which the label gets printed: you will need to alter the field format for both `labelalpha` and `extraalpha`. So, for instance, for bold labels you want

```
\DeclareFieldFormat{labelalpha}{\mkbibbold{#1}}
\DeclareFieldFormat{extraalpha}{\mkbibbold{#1}}
```

Author/Year Styles

Author/year styles set citations in a form such as (Bloggs 1982). They are common in many humanities subjects. Where an author has more than one work in the same year, alphabetic labels are added to make it clear which work is which (Bloggs 1982a, Bloggs 1982b). Labels are usually placed directly in the text, not in footnotes. The bibliography is necessary in order to provide detailed bibliographical information that is lacking in the text: it is usually organised so as to put author and year first in each entry, where it can easily be found.

Cotton, Frank Albert et al. (1999). *Advanced inorganic chemistry*. 6th ed. Chichester: Wiley.
 Reese, Trevor R. (1958). "Georgia in Anglo-Spanish Diplomacy, 1736–1739". In: *William and Mary Quarterly*. 3rd ser. 15, pp. 168–190.

Figure 4.4: Author/year bibliography

The standard Biblatex styles offer a small selection of different author/year citation styles. (Among the other available styles, the APA style, and the Chicago style if loaded with the `authordate` option² provide very fully developed author/year styles, which might well be preferable to the standard styles for many people.)

The essential questions you need to ask in order to choose a style are:

- Do You want citations *compressed*? In other words, if you cite (in a single citation) two works by the same author, do you want to see 'Doe 1982; Doe 1983' (uncompressed) or 'Doe 1982, 1983' (compressed).
- Do you want to use 'ibid' for repeated citations of the same source? (In other words, should successive citations of a given work produce '(Bloggs 1982) ... (ibid., p. 100)' instead of '(Bloggs 1982) ... (Blogs 1982, p. 100)'.

Depending on the answer to those questions, load the appropriate style as shown in table 4.4. (If you want multiple citations sorted but not compressed, you can set the option `sortcites` while loading an uncompressed style.)

BESIDES SELECTING from the range of styles, there's not much room for simple customization; but there are three things you may wish to adjust.

First, the number of names that get printed. By default, up to three names will be printed, both in the citations and in the bibliography. If there are more than three names, they will be abbreviated to one

² The Chicago style is loaded with `\usepackage[...]{biblatex-chicago}` rather than `\usepackage[style=chicago]{biblatex}`.

	compress	ibid
authoryear		
authoryear-comp	•	
authoryear-ibid		•
authoryear-icomp	•	•

Table 4.4: Author/Year styles

► See further, p 87

name ‘et al’ — again, both in the citations and the bibliography. So, for instance, the entry:

```
@book{companion,
  author = {Goossens, Michel and Mittelbach, Frank
            and Samarin, Alexander},
  title = {The LaTeX Companion},
  date  = {1994},
  ...
}
```

will produce ‘Goossens, Mittelbach, and Samarin 1994’; but if one additional author were added, the citation would become ‘Goossens et al’.

This behaviour can be altered, and it can be altered separately for references in the text and those in the bibliography list. If you prefer shorter lists of names, then use the option `maxnames`: if `maxnames=1` then no more than one name will be used, and so forth. You can, however, still have more than one name in the bibliography, because you can set `maxbibnames` to a different number. So if, for instance, `maxnames=1` but `maxbibnames=3`, then the in-text citations will read ‘Goossens et al’, but the bibliography will still read ‘Goossens, Mittelbach, and Samarin ...’.

► For further detail, see p 87

A second adjustment you might want to make is to punctuation:

► See generally, chapter 10.

- You might change the punctuation between author and the year. By default it is a space; one way to change it is by renewing the `\nameyear delim`. For instance, if you wanted to place a colon, rather than a comma, you would

► For more information about delimiters, and some more complex ways of handling them, see *Manual*, § 3.10.2

```
\renewcommand{\nameyear delim}{\addcolon\space}
```

and citations would become ‘Author: 1980’, and so forth. Another way of changing it uses the command `\DeclareDelimFormat`. So the same result is achieved with

► *Manual* 3.10.2

```
\DeclareDelimFormat{nameyear delim}{\addcolon\space}
```

Note that if you used the `\DeclareDelimFormat` method, the name of the delimiter is not preceded by a backslash.

- You can change the punctuation between multiple citations, by changing `\multicitedelim`, either using `\renewcommand` or using `\DeclareDelimFormat`.

The third set of adjustments that you might like to make arise only if you are using one of the style versions that makes use of ‘ibid’. You

can control the circumstances in which *ibid* is used by setting the `ibidtracker` option. There are five possible settings for `ibidtracker` described in the Biblatex documentation; but in my experience only three of them are useful

`ibidtracker=false` Don't ever use *ibid*.

`ibidtracker=constrict` Use *ibid*, but be cautious to avoid ambiguity. Footnotes and the main text are tracked separately, and *ibid* is used only if the reference will certainly be unambiguous.

`ibidtracker=true` The sloppiest use of *ibid*: use *ibid* whenever the previous reference was to the work in question, without tracking text and footnotes separately.

In most cases, the use of any setting sloppier than `constrict` seems questionable (and this is the default setting).

Author/Title Styles

The various author/title styles all use the author's name as the primary citation label. Depending on the particular style, a title is either always added, or is added only where it seems necessary to disambiguate the citation (because more than one work is being cited by the same author). A bibliography remains essential to the style, and will look like figure 4.5. However, the author/title styles lurk especially close to the borderline between labelled and fully bibliographical styles, and are appropriate for use either in footnotes or in running text. (In fact, the default styles which come with Biblatex differ as to whether they place citations generated with the `\autocite` command in text or footnotes.)

Cotton, Frank Albert et al. *Advanced inorganic chemistry*. 6th ed. Chichester: Wiley, 1999.

Reese, Trevor R. "Georgia in Anglo-Spanish Diplomacy, 1736–1739". In: *William and Mary Quarterly*. 3rd ser. 15 (1958), pp. 168–190.

Figure 4.5: Author/title bibliography

Simple as this may seem, there are actually a large number of standard styles dealing with this. They can however, be selected by answering three questions:

1. Do you want citations *sorted and compressed*. Suppose you have two works by Joseph Bloggs, his *First Book* and his *Second Book*, and one by John Smith, his *Book*. If you `\cite{bloggs1,smith,bloggs2}` an uncompressed style will give you 'Bloggs, *First Book*; Smith, *Book*; Bloggs, *Second Book*'. A compressed style will give you 'Bloggs, *First Book, Second Book*; Smith, *Book*'.

2. Do you want citations to be *terse*. If you use terse citations, then titles will be included only if they are required in order uniquely to identify the source. So, in the above example, titles will be included for Bloggs (because you are citing two works by him), but not for Smith, because you are citing only one. So `\cite{bloggs1, bloggs2, smith}` will give you ‘Bloggs, *First Book*; Bloggs, *Second Book*; Smith’.
3. If you cite the same work successively, do you want to use ‘*ibid*’. If you do, then

Some `\parencite{smith}` have pointed out that this is an unnecessarily complex range of options `\parencite[100]{smith}`.

will give you

Some (Smith, *Book*) have pointed out that this is an unnecessarily complex range of options (*ibid*, p. 100).

Based on the answer to these questions, you can choose from among the various author/title styles that are available. See table 4.5.³ (If you want multiple citations sorted but not compressed, you can use an uncompressed style but set the `sortcites` option.)

CUSTOMIZATION is not often required, but resembles that for the author/year styles.

- Customization of the number of names printed in citations and the bibliography is achieved using `maxnames` and `maxbibnames`, just as described above.
- The separator between author and title is usually a comma. To change it, redefine `\nametitledelim`. For instance, to change it to a colon,

`\renewcommand{\nametitledelim}{\addcolon\space}`

- The delimiter between multiple citations is usually a semicolon. To change it, redefine `\multicitedelim`.
- If you are using *ibid*, you can configure it using the options given above.

APART FROM THE STANDARD STYLES that come with Biblatex, there is a very sophisticated, and highly configurable, style written by Dominik Waßenhoven, the `author-title-dw` style, which is part of the `biblatex-dw` style package.

	compress	terse	ibid
author-title			
author-title-comp	•		
author-title-terse		•	
author-title-ibid			•
author-title-icomp	•		•
author-title-tcomp	•	•	
author-title-ticomp	•	•	•

Table 4.5: Author/title styles

³ The one missing style seems to be the combination of terse citations with *ibid*.

► And see chapter 10

► Page 46

Verbose styles

The essence of verbose or bibliographical citation styles is that they include information which is strictly bibliographical in the citations themselves. Because of this, they are principally designed to be used in footnotes — the information they provide would clutter text up much too much. And, quite often, in order to avoid excessive repetition of the same data, these styles make use of a variety of abbreviations or shorthands, such as *ibid.*, *op. cit.*, *loc. cit.*, *supra* and so on, though exactly which of these are used and how varies quite a bit from style to style.

Because these styles place the full bibliographical data in the text, the bibliography is more of an adjunct than an essential reference tool. (It actually looks, in the standard styles, just like the author/title bibliography in figure 4.5.)

Standard Bibl_{at}ex offers no fewer than seven different flavours of verbose citation style. For practical use, there should be added to these the footnote-dw style from Dominik Waßenhoven’s bibl_{at}ex-dw bundle, the bibl_{at}ex-chicago package, with option notes and the oscola style — all of which build on the standard styles to handle quite complex citations. In many cases these may be found more convenient, though the standard styles are definitely the right place to start if you are looking to produce a custom style of your own.

As in the author/year and author/title styles, the essential choice from the menu of verbose styles can be made by asking a few simple questions.

1. How do you want to handle repeat citations. The first time you cite a source, all the verbose styles will give you a full citation. The second time, all will give you a shortened citation. But do you want that shortened citation to be simply a short name? Or do you want a cross-reference added to the note in which the source was first cited?
2. Do you want to use *ibid* where two immediately sequential citations are to the same source?
3. Do you want to use additional scholarly abbreviations such as *op cit* and *loc cit*?

The standard verbose styles are, I think, not especially usable as they stand; but they form an excellent starting point for customization.

ONCE THAT CHOICE HAS BEEN MADE there is little scope for customization. Or rather, there is such *vast* scope for customization of every aspect of the citation — much of which requires customization of actual bibliography drivers — that it lies well outside the ambit of this chapter.

	first citation	subsequent	repeated
verbose	Frank Albert Cotton et al. <i>Advanced inorganic chemistry</i> . 6th ed. Chichester: Wiley, 1999	Cotton et al., <i>Advanced inorganic chemistry</i>	Cotton et al., <i>Advanced inorganic chemistry</i>
verbose-ibid	Frank Albert Cotton et al. <i>Advanced inorganic chemistry</i> . 6th ed. Chichester: Wiley, 1999	Cotton et al., <i>Advanced inorganic chemistry</i>	Ibid.
verbose-note	Frank Albert Cotton et al. <i>Advanced inorganic chemistry</i> . 6th ed. Chichester: Wiley, 1999	Cotton et al., <i>Advanced inorganic chemistry</i> , see n. ??	Cotton et al., <i>Advanced inorganic chemistry</i> , see n. ??
verbose-inote	Frank Albert Cotton et al. <i>Advanced inorganic chemistry</i> . 6th ed. Chichester: Wiley, 1999	Cotton et al., <i>Advanced inorganic chemistry</i> , see n. ??	Ibid.
verbose-trad1	Frank Albert Cotton et al. <i>Advanced inorganic chemistry</i> . 6th ed. Chichester: Wiley, 1999	Cotton et al., op. cit.	Ibid.
verbose-trad2	Frank Albert Cotton et al. <i>Advanced inorganic chemistry</i> . 6th ed. Chichester: Wiley, 1999	Cotton et al., <i>Advanced inorganic chemistry</i> , op. cit.	Ibid.

Table 4.6: Various verbose styles.

However there are some options which can be adjusted: the behaviour of *ibid* (see above), for example, or sorting schemes (see chapter 8).

► p 45

5

Citation Commands

The heart of any citation package, from the user’s point of view, is the citation commands. These are what you use most. Biblatex offers a number of different commands. They follow a consistent pattern; so once you’ve mastered one it’s quite easy to grasp the others. Their exact output depends on the citation style you are using.

Let’s start not with the commands themselves, but by looking at some citations. Consider the following:

See also [1]
 (Jones 1990, p. 24)
 cf [JJ90, ch. 3]
 see Jones, *Combinatorial Aesthetics* (London: Pubco, 1990)
 ibid. 202

There’s a common pattern behind the obvious differences. Each citation consists of up to three parts: an (optional) ‘signal’ that introduces it; the citation itself, which directly or indirectly identifies a source; and an (optional) ‘pinpoint’ which locates a particular part of the cited work.

The citation commands in Biblatex follow the same pattern: they have one mandatory argument, which specifies the ‘key’ of the source cited, and two optional arguments. The primary optional argument is called the postnote and gives the ‘pinpoint’, and the secondary optional argument is called the prenote and specifies the ‘signal’.

	prenote	postnote	key
<code>\cite[See also][]{jones}</code>	See also		jones
<code>\cite[24]{jones}</code>		p. 24	jones
<code>\cite[cf.][ch. 3]{jones}</code>	cf.	ch. 3	jones
<code>\cite[see][]{jones}</code>	see		jones

Notice that if there is only one optional argument it is assumed to be the postnote; but if there are two, then the first optional argument

signal	citation	pinpoint
See also	[1] Jones 1990	p. 24
cf.	JJ90	ch. 3
see	Jones ... ibid.	202

Table 5.1: The structure of citations

Table 5.2: Citation command with prenotes and postnotes

is the prenote. If we want a prenote but no postnote, as in the first and last examples, we have to leave the second argument blank. But if we want a postnote and no prenote, we just use one optional argument. All the citation commands follow this format.

The Basic Five

There are five basic citation commands: `\cite`, `\footcite`, `\parencite`, `\autocite` and `\textcite`. Exactly what each one does depends on the citation style. But the general idea of the circumstances in which you would use each is more or less the same.

\cite. The `\cite` command is “basic”. It simply prints the particular style’s idea of a simple citation. So, for instance, with a numeric style, `\cite` prints [1], and with a verbose style it prints full bibliographical details (or *ibid.*, or *supra n.* . . . , if that would be correct).

\footcite and \parencite. The `\footcite` command tries to put its citation in a footnote. So `\footcite{jones}` means the same thing as `\footnote{\cite{jones}.}` — syntactic sugar. The `\parencite` command does a similar thing, but it wraps its citation in parentheses, so that it is equivalent to `(\cite{jones})`.

\autocite. This is all very well. But what if you’re not sure whether you should use `\cite` or `\footcite` or `\parencite`. Suppose you write a paper, expecting to use numerical citations; but then you decide to use verbose citations instead. Now you have to go through and convert all your `\cites` to `\footcites`. The `\autocite` command is intended to avoid that. It will use whatever citation command is most appropriate to the style selected. So if you use `\autocite`, you can (try to) change all your citations from in-text citations to, say, footnotes simply by changing the style.

It does another thing that is, as the kids say, cool. It will shift punctuation. It needs to be able to do this because where a citation appears relative to punctuation may vary. In English and American practice, for instance, footnotes appear after punctuation. So whereas a correct numerical citation would be ‘blah [1].’ a correct verbose one would be ‘blah.¹’ With ‘`\autocite{jones}.`’ Biblatex will move punctuation from after the citation command to put the footnote marker in the right place.

There are, however, limitations. The `\autocite` command can cope well with citations at the end of sentences; but if citations are entangled in the text too much, the result will not be satisfactory. A sentence like

Jones (1990) and Smith (1991) ‘independently concluded’ (Doe 2002, p. 19) that foos should not be permitted to bar.

will probably require some manual intervention.

`\textcite`. Finally there is `\textcite`. It’s not uncommon to have sentences like

... as Jones (1990) remarks ...

where the citation is ‘woven’ into the text of the sentence.

In numerical and verbose styles this can be done manually:

as Jones `\cite{jones}` remarks → as Jones [1] remarks

But that’s a bit wordy, and in author/year styles, this trick won’t work:

as Jones `\parencite{jones}` remarks → as Jones (Jones 1990) remarks

`\textcite` is designed to deal with this. It offers a citation in a form appropriate for running text.

as `\textcite{jones}` remarks → as Jones (1990) remarks

Although mainly intended for author/year styles, `\textcite` works rationally in any style; in verbose styles, the proper form involves printing something in the text (such as ‘Jones’) and adding a footnote; if necessary punctuation will be moved for this purpose, so that

... as `\textcite[11]{Jones}`, presciently, remarked ...

will produce

...as Jones¹ presciently, remarked

The comma is moved so that it appears before the footnote.

Summary

Table 5.3 shows how the various citation commands operate with the various different standard styles.

	<code>\cite[9]{jones}</code>	<code>\footcite[9]{jones}</code>	<code>\parencite[9]{jones}</code>	<code>\autocite[9]{jones}</code>	as <code>\textcite[9]{jones}</code> says
numeric	[1, p. 9]	*1, p. 9.	[1, p. 9]	[1, p. 9]	as Jones [1, p. 9] says
alphabetic	[Jon94, p. 9]	*Jon94, p. 9	[Jon94, p. 9]	[Jon94, p. 9]	as Jones [Jon94, p. 9] says
authoryear	Jones 1994, p. 9	*Jones 1994, p. 9	(Jones 1994, p. 9)	(Jones 1994, p. 9)	as Jones (1994, p. 9) says
authortitle	Jones, “Title”, p. 9	*Jones, “Title”, p. 9	(Jones, “Title”, p. 9)	*Jones, “Title”, p. 9	as Jones (“Title”, p. 9) says
verbose	Jones. “Title”. In: <i>Journal</i> 100 (1994), p. 7, p. 9	*Jones. “Title”. In: <i>Journal</i> 100 (1994), p. 7, p. 9	(Jones. “Title”. In: <i>Journal</i> 100 (1994), p. 7, p. 9)	Jones. “Title”. In: <i>Journal</i> 100 (1994), p. 7, p. 9	as Jones* says
					*Jones. “Title”. In: <i>Journal</i> 100 (1994), p. 7, p. 9.

Table 5.3: Common citation commands in standard styles

Capital Forms

Each of `\cite`, `\parencite` and `\autocite` have ‘capital’ forms (`\Cite`, etc) which are appropriate for use where a capital would be required. You might wonder why, since most citations already begin with a capital, or with a number. The only time capitalisation might be needed is if the citation begins with something like ‘von’, and you want it capitalised, or if the footnote reads ‘ibid’ and you need ‘Ibid’. There is no special capital form of `\footcite` because it always assumes that the citation will begin a sentence so that a capital may be appropriate.

Specific to Numerical Styles

For numerical styles, the command `\supercite` prints citations as superscript numbers. It’s rather unusual to want both regular full-size labels and superscript labels in the same document; but if you do you can mix `\cite` and `\supercite` as you wish. Generally speaking you want one or the other. For this, the safest course is to use the option `autocite=super`, and use `\autocite`.

There is a limitation to `\supercite`. It doesn’t make sense to try to cram signals and pinpoints into superscript numbers. So `\supercite` will discard prenote and postnote parts

```
\supercite[see][25]{jones} → 1
```

If this happens, a warning will be given.

Using particular fields

There are a number of citation commands that are not intended to print complete citations, but to pull potentially useful snippets out of a source record, and treat it as a citation. They are mostly, though not exclusively, useful in author/title and author/year styles, where citations are often quite tightly woven into the text.

Author or title alone. The commands `\citeauthor` and `\Citeauthor` print the author’s name (the latter with capitals forced, even if they would otherwise not be used). The commands `\citetitle` and `\citetitle*` print the title: abbreviated if possible in the regular form, but always in full in the starred form.

Year or date. The commands `\citeyear` and `\citeyear*`, and `\citedate` and `\citedate*`, print year or date. The difference is that whereas the unstarred versions just print the calendar year or date, the starred

```
\citeauthor{key} published his
roman a clef (\citeyear*{key}) in
\citeyear{key} → Locke published his
roman a clef (1905b) in 1905
```

versions print any extra label that is added to that for the particular work. So if you had, say, five works by a particular author in one year, `\citeyear{prolific86c}` would print ‘1986’, whereas `\citeyear{prolific86c}` would print ‘1986c’.

URLs. The `\citeurl` command will print any url.

Others. Apart from these commands, which the standard styles already provide, it is possible to construct your own ‘partial’ commands, using `\citename`, `\citelist` and `\citefield`. There are three commands because Biblatex distinguishes between name fields (like author and editor), list fields (like institution and publisher) and other fields. The commands are not really suitable or intended for direct use: they would need to be ‘wrapped’ in some more user-friendly command for use in a document.

► Manual § 3.8.7

Entry sets

In some disciplines it is common to have a single numeric citation refer to a group of sources. This is supported in Biblatex by the entry-set.

► Manual § 3.12.5

There are two ways to define such a set. One way is to define them as a key in the .bib file. Thus, for instance, your .bib file might define

```
@set{set1,
  entryset={augustine, cotton}}
```

You can then use `\cite{set1}` to get output as in figure 5.1.

This is all explained in [1].

- [1] Robert L. Augustine. *Heterogeneous catalysis for the synthetic chemist*. New York: Marcel Dekker, 1995; Frank Albert Cotton et al. *Advanced inorganic chemistry*. 6th ed. Chichester: Wiley, 1999.

Figure 5.1: An entryset

The alternative, and usually more convenient approach, is to define such sets ‘on the fly’ in each document. This can be done using the command

```
\defbibentryset{<set-key>}{<key1 ... keyn>}
```

so that

```
\defbibentryset{set1}{augustine, cotton}
```

will produce the same output as figure 5.1.

Pinpoints

It usually makes sense to give pinpoint citations to a particular work primarily by reference to one type of marker. So, for instance, books are normally cited by page. How such references are styled depend on the particular scheme. Some schemes want ‘p. 23’, others want ‘p 23’, others just want ‘23’. To save effort, Biblatex will insert appropriate default labels. So if you just type `\cite[23]{key}`, Biblatex will produce ‘p. 23’ (or whatever is appropriate to your style). It works also for multiple pages: `\cite[22, 23]{key}` will print ‘pp. 22, 23’ and `\cite[22-25]{key}` will print ‘pp. 22–25’, and do forth.

What if you want some different prefix? If the problem is with a particular citation, you can just enter the prefix manually. For instance, if you want to cite chapter 3, just type `\cite[ch.~3]{key}`. Since Biblatex only adds a prefix where you have provided a reference which consists only of numbers, your postnote will be printed as you have entered it.

THIS IS NEAT AND TIDY. But, as so often there are special cases and customisations.

No prefixes You don’t want prefixes, even when you enter a purely numerical postnote: you want `\cite[22]{key}` to produce ‘*key*, 22’. There are a number of ways to achieve this, depending on how generally you want to disable the use of prefixes. For a single citation, start the postnote with the special command `\nopp`. This is most appropriate where you have a single citation that is giving you problems.

`\cite[\nopp22]{key} → key, 22`

To stop all additions for a particular source, set that source’s pagination to ‘none’. This is most appropriate when you generally want to use prefixes, but you have a particular source where they are not required.

`@book{key,...
 pagination = {none}}
 \cite[22]{key} → key, 22`

To stop all additions, for every source:

`\DeclareFieldFormat{postnote}{#1}`

This is most appropriate when you are customising Biblatex for a style which does not use any prefixes in general.

Forcing prefixes You want to enter a postnote with a non-numerical character, and still get a prefix added. For instance, you have a source with pages which have letters, like ‘1234a’. Again, there are a number of solutions.

The easiest thing to do in such cases is simply to type the prefix yourself: `\cite[p.~1234a]{key}`. But if you think it’s worth getting

`\cite[p.~12a]{key} → key, p. 12a`

Biblatex to do it for you, you can use `\pno` (for a single page) and `\ppno` (for multiple pages). (In theory, the `\pno` and `\ppno` commands could be ‘better’ because you could change the style of prefixes and they would adjust automatically. In the real world, there’s probably not much practical advantage.

Latin gadgets for multiple pages There is one special common case. Suppose you want ‘p. 20 ff.’ or ‘p. 20 et seq.’ You could of course enter the whole postnote manually. But you can also use the special commands `\psq` and `\psqq` to add the indication. The advantage is that the particular phrase to refer to subsequent page(s) can then be set consistently. You should not put any space between these commands and the number: `\cite[20\psqq]{key}`.

By default, in the standard styles, `\psq` gives ‘sq.’ and `\psqq` gives ‘sqq.’ To redefine them, redefine the bibliography strings `sequens` (for a single following page), and `sequentes` (for more than one). For instance, to use ‘et seq.’ for both, one would (assuming we are using American English)

```
\DefineBibliographyStrings{american}{%
  sequens = {et seq\adddot},
  sequentes = {et seq\adddot}}
```

The space before this addition is added by a command called `\sqspace`. Some citation styles don’t add any space at this point, preferring ‘1of.’ to ‘1o f.’ If you don’t want any space, redefine this command:

```
\renewcommand{\sqspace}{{}}
```

Correcting Biblatex’s guesses Biblatex tries to guess whether to add ‘p.’ or ‘pp.’ by working out whether you have cited a single page or a range of pages. But occasionally the form of a citation might confuse Biblatex, so that it adds ‘p.’ when you need ‘pp.’, or vice versa. In such cases, either type the whole postnote yourself, or use the `\pno` or `\ppno` commands, as explained above.

You want something ... but not ‘pp.’ Some sources don’t use pages for pinpoint citations. For instance, suppose you are citing a work which is normally referred to by section, not page. In such a case, you can still use the convenience of having automatic prefixes. What you need to do is to set the pagination field of the source in the `.bib` file to ‘section’. A list of the types of pagination recognised, and their results, is shown in table 5.4.

That’s fine so long as Biblatex recognises the pagination type. But if it doesn’t, you will get odd results. For instance, suppose we want references in the form ‘Art. x’, so that `\cite[2]{key}` gives us ‘Art.

```
\cite[\pno 12a]{key} → ⟨key⟩, p. 12a
\cite[\ppno 12a--b]{key} → ⟨key⟩,
pp. 12a–b
```

```
\cite[20\psqq]{key} → ⟨key⟩, pp. 20
sqq.
```

pagination	singular	plural
page	p.	pp.
column	col.	cols.
section	§	§§
paragraph	par.	pars.
verse	v.	vv.
line	l.	ll.
none		

Table 5.4: Standard values for pagination

2'. If we enter the pagination as article, however, we just get 'p.', because article is not a pagination type that Biblalex knows.

How can we 'teach' Biblalex a new pagination type? In this example, we simply need to define two bibliography strings—article and articles:

```
\NewBibliographyString{article, articles}
\DefineBibliographyStrings{american}{% or your language
  article = {Art\adddot}
  articles = {Arts\adddot}}
```

Now all will work as intended.

Modifying standard prefixes To change the standard prefixes, you can use essentially the same trick – only you don't need to define the relevant bibliography strings because they already exist. So, for instance, if you wanted the abbreviation for page to be "pg", you could redefine the relevant strings as follows

```
\DefineBibliographyStrings{american}{% or your language
  page = {pg},
  pages = {pgs}}
```

Multiple citations

You've probably worked out by now that even the standard citation commands will handle multiple citations: `\cite{key1, key2}` will produce '[1, 2]' or 'Jones 1990; Smith 2010', or whatever is in keeping with the style you have chosen.

This is fine; but it doesn't handle pre- and postnotes very nicely. If we enter `\cite[see][11]{key1, key2}` we get something '[see 1; 2, p. 11]' — which will only make sense if the page reference is to the last work cited. For this purpose, Biblalex offers a selection of commands tailored for multiple citations.

The pattern is set by the `\cites` command. If we type

```
\cites[See][10]{key1}[cf][20]{key2}[30]{key3}
```

we would get (in a numeric style)

[see 1, p. 10; cf 2, p. 20; 3, p. 30]

or (in an alphabetic style)

see Smith 1990, p. 10; cf Jones 2000, p. 20; Bloggs 2010, p. 30

regular	multi-cite version
<code>\cite</code>	<code>\cites</code>
<code>\footcite</code>	<code>\footcites</code>
<code>\parencite</code>	<code>\parencites</code>
<code>\autocite</code>	<code>\autocites</code>
<code>\textcite</code>	<code>\textcites</code>
<code>\Cite</code>	<code>\Cites</code>
<code>\Parencite</code>	<code>\Parencites</code>
<code>\Autocite</code>	<code>\Autocites</code>
<code>\Textcite</code>	<code>\Textcites</code>

Table 5.5: Multiple citation commands

It's not hard to see what's happening here: basically Biblatex is interpreting `\cites` as a set of individual `\cite` commands, each with its own pre- and/or postnote, but also understanding that they form a unit (so, for instance, the numeric style places them in one set of brackets).

```
\cites [see][10]{key1} [cf][20]{key2} [30]{key3}
```

To make things even more exciting, you can also have a pre- or postnote for the *entire* multiple citation: you just place them (before anything else) in parentheses:

```
\cites(see:)(among others)[10]key 1[20]key 2
```

giving

see: Smith 1990, p. 10; Jones 2000, p. 20, among others

But there's one catch. As it scans forwards, the `\cites` command will be 'tricked' if it sees a square bracket (`[`) or a brace (`{`) which is not intended to start a citation. For instance, in:

```
\cites [10]{key 1} [20]{key2} [‘the principal works’]
```

a human can quickly see that the blue portion is not another citation, but some text in brackets. But Biblatex won't appreciate this (a space is not enough), and will complain that you have a defective citation. The trick in such cases is to put `\relax` at the end of the list:

```
\cites[10]{key1}[20]{key2}\relax [‘the principal works’]
```

All the main citation commands (and their capitalised forms) have multiple versions, consistently named: see table 5.5.

Citation commands that ... don't

Finally we have to consider a small family of odd citation commands. Normally the point of including a citation is to have *some sort* of information about the source in question. It might, therefore, seem odd that there are citation commands which don't print anything. But they are sometimes needed. For instance, you might want to include a work in the bibliography without actually printing any citation in the text. To do that you can use the `\nocite` command. So `\nocite{angledkey}` will place *key* in the bibliography without citing it. You can also use `\nocite{*}`, which will put *every source* in your bibliography file into the bibliography.

Another occasionally useful command is `\mancite`. You use this command to tell Biblatex that you have manually cited a work. You might like to do this if you want to weave the citation into the text in

a way that is too complex to achieve using the various citation commands. It may be worth doing, because it will make sure that Biblatex keeps track of internal housekeeping that matters. For instance, suppose you had the following:

```
... is clear \parencite{jones}. Smith's 'rebuttal' (see
above) hardly meets this point; but this is not Jones's
main objection \parencite[45]{jones}. ...
```

If you were using a citation style which used 'ibid', the second reference to Jones would be printed as 'ibid', since as far as Biblatex is concerned there are two consecutive citations to it. But the sentence describing Smith's rebuttal is a sort of implicit citation, and you might think 'ibid' ambiguous in this context. So you could type

```
... is clear \parencite{jones}. Smith's 'rebuttal' (see
above)\mancite{smith} hardly meets this point; but this is
not Jones's main objection \parencite[45]{jones}. ...
```

Generally, however, it's better to avoid such implicit citations.

6

The Bibliography

Citations written, we come to the actual printing of the bibliography. The magic command is:

`\printbibliography`

► *Manual* § 3.7.7

► *Manual* § 3.7.2

And, for simple cases, it may even be as easy as that! But there are quite a number of refinements.

The refinements can be classified as follows:

- Changes to the way the bibliography is headed: what is it called? Is it treated as if it were a new chapter, or a new section? How is the heading printed? Is it included in the table of contents?
- Changes to the appearance of the bibliography items: what font are they printed in? Are they indented? How? How much?
- Changes to the way the bibliography is sorted.
- Selection of the material to be printed in the bibliography: are all sources cited, or only some? How are they selected?
- Provisions for creating multiple bibliographies, such as different bibliographies for different chapters, or different bibliographies for different topics.

The last two items overlap a bit, because in one sense producing multiple bibliographies is a matter of selecting what works go in which bibliography. It's simpler, however, to defer most discussion of this to a chapter of its own. We are also going to leave sorting for a separate discussion. So this chapter is really going to concentrate on the headings, the physical format of the bibliography.

► Chapter 7

Headings

In a \LaTeX document, a heading is more than just its words: a heading command (such as `\chapter` or `\section`) doesn't just print its

text, formatted a certain way. It may add a number, and advance a counter. It may change what gets printed in the running headers or footers.

Let's start with the actual title. A default¹ will be set by the heading option, described below. The easiest way to change it is using the title option in the `\printbibliography` command.

So, for instance, if we wanted to call our bibliography 'List of Sources', we could put the following:

```
\printbibliography[title={List of Sources}]
```

This is fairly straightforward. Beyond that, some degree of complexity lurks.

It's all fairly easy if you want to have the bibliography title treated as if it were a chapter heading (for the book and report classes) or a section heading (for the article class). This is, after all, the ordinary case, and Biblatex then provides you with the following options:

- If you want the heading inserted, but without any number and without being added to the table of contents, you're all set. This is what Biblatex will do by default.
- If you want the heading added to the table of contents, but you don't want it numbered, then add the option `heading=bibintoc` to your `\printbibliography` command.
- If you want the heading numbered and added to the table of contents, then add the option `heading=bibnumbered` to your `\printbibliography` command.

→ **References**
(not in table of contents)

→ **References**
(in table of contents)

→ **6. References**
(in table of contents)

It's also easy enough if you want the bibliography dealt with at *one level below* the default: in other words, you want it treated as a section (in the book or report classes) or as a sub-section (in the article class):

- To have just the heading (no table of contents or numbering), choose the option `heading=subbibliography`.
- To have the heading included in the table of contents, but not numbered, choose the option `heading=subbibintoc`.
- To have the heading numbered and included in the table of contents, choose the option `heading=subbibnumbered`.

Finally (and perhaps easiest of all), if you want no heading at all, just choose `heading=none`.

THESE ARE ALL THE OPTIONS that Biblatex gives you by default. Combined with `title=...` (to override the default titles) this will

¹ The default is 'Bibliography' for books and articles, and 'References' for articles.

usually suffice. If, however, you want to define a special heading for your bibliography, you will need to proceed as follows:

► *Manual* § 3.6.7

1. Define a bibliography heading style using

```
\defbibheading{⟨name⟩}[⟨title⟩]{⟨definition⟩}
```

The *⟨name⟩* is a name you will use, as an option passed to `\printbibliography`, to refer to the type of heading that you have defined; it can be anything you choose. The *⟨title⟩* is the default title for your heading. The definition part is a complete set of commands to create a heading and anything (such as the marking of headers and footers, table of contents, and so forth) that needs to go with it. It takes the form of a macro definition which assumes it will be passed the title of the bibliography as its single argument, #1.

2. Use the name of the heading you have defined as the heading option to your `\printbibliography` command.

So, for instance, suppose we wanted to set up our bibliography to be formatted as a subsection, but without any number; we also want it added to the table of contents, and we want it to mark both recto and verso pages. The default title is to be ‘Works Cited’:

```
\defbibheading{myheading}[Works Cited]{%
  \subsection*{#1}%
  \addcontentsline{toc}{subsection}{#1}%
  \markboth{#1}{#1}}
```

There’s one tiny trick to bibliography headings. It’s normally best to do all your customization in the preamble of your document, before you (or L^AT_EX) gets to `\begin{document}`. But for bibliography headings (and title, and notes) it’s actually better to set them up after `\begin{document}`.

Bibliography appearance

The appearance of a bibliography is largely controlled by a pre-defined bibliography environment. You can change the default by defining a modified bibliography environment of your own, and then passing the name of that environment as

```
\printbibliography[env=⟨your environment⟩]
```

The environment needs to be set up in advance, for that purpose, you use the command `\defbibenvironment`:

```
\defbibenvironment{<name>}{<start>}{<end>}{<per-item>}
```

The way this works is as follows:

`<name>` is the name you choose to refer to this style of environment: it's what you will use with the option `env=` to typeset a bibliography using this environment.

`<start>` is code to be executed at the beginning of the bibliography, before any entries are printed. It's customary (though not essential) to set a bibliography using a list environment of some sort, and this lets you set that up.

`<item>` is code to be executed at the beginning of each *entry* in the bibliography: for instance `\item`.

`<end>` is code to be executed at the end of the bibliography, for instance to end the environment you began with `<start>`.

Let's give a practical example. People writing CVs sometimes want to number the items while using a non-numeric bibliography style, such as verbose or author-year. Suppose you loaded the `authoryear` style, but then defined a bibliography environment as follows:

```
\defbibenvironment{myenv}
  {\begin{enumerate}}
  {\end{enumerate}}
  {\item}
```

Now, using

```
\printbibliography[env=myenv,heading=myheading]
```

we get

Works Cited

1. Prolific, Pete (2013a). "Keeping it Coming". In: *Obscure Studies* 66, p. 104.
2. — (2013b). "Pumping it Out". In: *Unread Papers* 23, p. 367.

Figure 6.1: Customized bibliography environment

Other lists and indexing

If you use 'shorthands' to define abbreviations for particular works, you can print these with the command:

► See p 24


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

You will almost certainly want to provide a suitable title using the option `title=...`. As the general form of the command suggests, it is in fact an instance of a general way of printing special forms of bibliographical list.

► *Manual* § 3.7.3

Biblatex also enables styles to permit the creation of an *index* of sources. The details are elaborate, and you will need to understand something about how L^AT_EX's indexing methods work. Some styles (such as *oscola*) make very extensive and complex use of this facility. But, as it is a rather advanced topic, the curious reader is referred to the manual, both for Biblatex and for any package that is being used.

► *Manual* § 3.1.1

Slightly simpler, however, than an index (which is a separate document), it is possible also with standard styles to print, in the bibliography, a list of the pages where the source in question was referred to. To do this, you pass Biblatex the option `backref=true`. There are various options for `backrefstyle` which will determine how back references are actually printed (for instance, when references in consecutive pages are compressed).

► *Manual* § 3.1.1

7

Multiple Bibliographies

We now come to consider the various methods of producing *multiple* bibliographies in a single TeX document. The detail can become overwhelming unless you first understand the logic, so that is where we will start.

There are two main patterns to split bibliographies:

1. Division that is determined by characteristics of the *source*, where some types of source get put in one bibliography and other types of source get put in another. The dividing line might be between primary and secondary sources. It might be between works about foology and everything else ('topic based'). It might be between the most important works and the rest. It might be between the works of Dr Drofnats and everyone else's works. But however the dividing-line is drawn, it is based on something about the underlying source, which is why I'm going to call these *source-based* divisions.
2. Division that is determined by the source's *citation*, and in particular by *where* the source is cited — so that, for instance, there are separate bibliographies for Chapter 1, Chapter 2, and so forth. In an 'extreme form' the bibliographies need to be completely self-contained; that's often needed, for instance, if a single document is being used to typeset papers by different authors into a collection or journal issue. I'm going to call these *document-based* divisions.

These categories are not, of course, mutually exclusive. You *could* have, in a single document, both subdivision by source (say into primary and secondary sources) and subdivision by location in the document.

Biblatex provides mechanisms to handle both types of division. It is designed to be comprehensive. Before Biblatex there were many competing packages (multibib, splitbib, chapterbib, bibunits, bibtopic) which provided these facilities with biblatex. These are *not* compatible with Biblatex; you cannot use them together — and you don't need to.

Source-based division

In principle, source-based division is simple. You provide some way for Biblatex to identify the different types of source, and then you print separate bibliographies for each. Although the most interesting features are those that determine how works are selected, it's convenient to start by explaining how the actual printing is handled.

```
\printbibliography[title="Primary Sources", <selector>]
\printbibliography[title="Secondary Literature", <selector>]
```

Usually in these cases, however, the separate bibliographies are seen as 'sub-bibliographies', in which case a little bit more work is done to get the formatting right:

```
\printbibheading \printbibliography [heading=subbibliography,
title="Primary Sources", filter=<selector>] \printbibliography
[heading=subbibliography, title="Secondary Literature",
filter=<selector>]
```

We use `\printbibheading` simply as a convenience to print a standard 'top level' heading for the whole bibliography, but then use the heading and title options when actually printing the bibliography. As far as Biblatex is concerned, a document can contain as many `\printbibliography` commands as you like, and each can contain as many or as few sources as you like. The real trick lies in how you teach Biblatex to distinguish between different sources. In all cases, the method used is to add an option to the `\printbibliography` command which tells Biblatex what filter to use.

To simplify somewhat there are three basic ways you can do that, and a convenient way of combining them. The basic ways involve: *type*, *keywords* and *categories*. The combining device is the *filter*, which enables complex assemblies of conditions to be defined and used simply.

How Biblatex chooses. All these distinctions are brought into operation using an option in the bibliography heading. These follow a standard pattern:

```
<selector> = <value>
```

(e.g. `type=book`) to include only books.

If you prefer to look on the negative side:

```
not<selector> = <value>
```

(e.g. `notttype=online`): exclude from the bibliography entries of this type). You can have multiple 'excluders', which are applied cumulatively: `[notttype=online, nottype=misc]` will *exclude* both online and misc entry types.

THE EASIEST TYPE OF CHOICE to build in to a bibliography is between different entry types. Every source necessarily has an entry type, and this can be a convenient way of producing some kinds of subdivided bibliography. For instance, if you wanted to distinguish between printed sources (the bibliography) and online sources (the ‘webography’) it might be sufficient to distinguish between. Here you can use:

- `type`, to include (only) particular entrytypes, e.g. `type=book` to include only books. You can only have *one* positive type selector. If you want to include, say book and article types, you will need to define a `bibfilter` for that purpose;
- `notype`, to exclude particular entrytypes, e.g. `notype=book` to exclude books. You can always have *more than one* excluder, without the need to define a `bibfilter`.
- `subtype`, to include particular entrysubtypes (some bibliographic styles make use of subtypes to further divide types, or to specialised types such as `misc`)
- `notsubtype`, to exclude particular entrysubtypes.

(The reason you can have multiple excluders, but not multiple includers, is that generally Biblatex treats conditions as cumulative. So (`notype=book` and `notype=article`) works to identify sources which are neither books nor articles. But (`type=book` and `type=article` ‘work’ to identify sources which are both books and articles. Inevitably, nothing matches.)

So, for example, we could build a separate ‘webography’ and ‘bibliography’ along these lines:

```
\printbibheading
\printbibliography[heading=subbibliography,
                    title="Webography",
                    type=online]
\printbibliography[heading=subbibliography,
                    title="Paperography",
                    nottype=online]
```

HOWEVER, TYPE OF SOURCE IS OFTEN NOT SUFFICIENT: many potentially interesting divisions (such as between primary and secondary sources, or by subject) cut across source types. The trouble here is that, for obvious reasons, Biblatex is not going to know about the underlying subject-matter of a source or whether it is primary or secondary and so forth unless you tell it. There are two ways to do this.

First, you can store the information *in the .bib file* using the *keywords* field, and then filter inclusion in a particular bibliography using the `keyword` and `notkeyword` instructions. So, for instance, you might mark every piece of primary literature using the keyword ‘primary’ in the .bib file. You could then distinguish between primary literature and secondary sources as follows:

```
\printbibliography[title = "Primary Sources",
                    keyword = primary]
\printbibliography[title = "Secondary Material",
                    notkeyword = primary]
```

That, however, is not very flexible. The alternative approach allows document-by-document flexibility. You can enter the information in the *document source* (.tex) file using the *categories* system. To do this:

► Manual § 3.12.4

- You first define a category in the preamble of the document, and then use a command to identify which sources are within a particular category. For instance, suppose you wanted to print ‘crucial’ works separately in some notes for your students, you would create that crucial category in your preamble:

```
\DeclareBibliographyCategory{crucial}
```

- You then add individual sources to this category anywhere in the document using `\addtocategory`, which takes two arguments: the category and a list of sources to be added to that category.

```
\addtocategory{crucial}{\langle key1 \rangle, ... \langle key2 \rangle}
```

You can have multiple keys and multiple `\addtocategory{}{}{}` commands as well.

- Then, when it comes to printing the bibliography, you control what is printed using `category=\langle category \rangle` and `notcategory=\langle category \rangle`.

```
\printbibliography[heading = "Key sources",
                    category = crucial]
\printbibliography[heading = "Other sources",
                    notcategory = crucial]
```

Both keywords and categories are ways of arbitrarily allocating sources to groups. The difference between them is that keywords are defined in the database, and categories in the document where the database is used. This should give you a hint about the best way of using them. A keyword should really say something permanently true about a source: they are a pretty reasonable candidate for dividing sources into primary and secondary, or original and translated, or

identifying their topic as foolery rather than barography. Categories need only be true for the particular document: they are a good candidate for specifying a source as ‘important’ in a set of student notes for instance, or for quick-and-dirty work where you don’t want to alter a database file permanently.

Combining criteria with user-defined filters

Suppose you want to combine filters in complicated ways: for instance you want a sub-bibliography which contains only crucial primary sources which are not online. For such purposes, one reaches for a user-defined filter. Such a filter is defined using

```
\defbibfilter{<name>}{<definition>}
```

This enables you to combine the type, subtype, keyword and category filters using logical operators (and, or, not) and parentheses for grouping. Having defined the filter, it can then be used to construct a bibliography with the option `filtername=<filter>`

So, for example, we could define the selection filter mentioned above as follows:

```
\defbibfilter{croffprim}{
  keyword=primary
  and category=crucial
  and not type=online}
```

And we could use it:

```
\printbibheading
. . .
\printbibliography[heading=subbibliography,
                    title="Crucial offline primary sources",
                    filter=croffprim]
```

Automation

Both keyword and category filters ultimately depend on direct user intervention, key-by-key. Is there any way to automate this? For instance, could you automatically assign (say) all the sources in a particular .bib file to a given category? Or could you assign all the works of a particular author?

The answer is that you often can do so. But it’s rather an advanced topic. Besides, the range of possible uses and circumstances is very wide. This section, therefore, is going to sketch some ideas, rather than provide a set of recipes.

- Define a low-level user-defined filter using `\defbibcheck`, which then uses the `check=<filter>` filter. For instance the Biblatex manual demonstrates using such a mechanism to examine the year of an entry and print a bibliography of ‘recent’ literature. ► Manual § 3.7.9
- Use a `sourcemap` to add keywords automatically. This could easily be done, say, for a particular file, or for a particular author’s work; you could then use a keyword filter. ► Manual § 4.5.3
- Use the `\AtEveryCitekey` hook to execute code at every citation to check data and decide whether to add that key to a category. ► Manual § 4.10.6
- Use the `\AtDataInput` to execute code as each entry is read in and decide whether to add it to a category. ► Manual § 4.10.6

Document-based division

Biblatex offers two main devices for producing bibliographies that are categorised based on where the citation is used in the LaTeX source file:

- A *refsection* is (nearly) a fully self-contained ‘unit’ which might have its own `.bib` files, and is expected to produce a basically self-contained bibliography. It is the obvious choice, for instance, if producing a collection of papers, each of which had a different author who had used his or her own `.bib` file, with its own keys.
- A *refsegment* is a less strongly-marked division, which represents a part of what is recognisably a larger work: a way of dividing a *single* bibliography up for convenience. It might be the right choice if you wanted to produce both summary bibliographies by chapter and a complete bibliography for the whole document.

The conceptual difference between the two is best understood if you imagine a bibliography style which uses numerical labels. If the same work is cited in two different *refsections* then it will be expected to appear in two different bibliographies with a *different* number: each bibliography is entirely self-contained, and the labels are unique within each but not between each. If, on the other hand, the same work is cited in two different *refsegments* then it will have a single, unique, label which will be common to both. The bibliography for each segment may print all and only the sources referred to in that segment; but although a source that is referred to in two different segments will be printed in both segments’ bibliographies, it will have the *same* label in each.

Another key difference is that whereas *refsegments* share the same .bib files, *refsections* need not. Some resources can be defined so as to be global, but individual refsections can also have their own ‘private’ resources. Why does this matter? Suppose that Dr Drofnats and Dr Frobwangler each writes a paper for a collection. In their field, the seminal work is Mangel-Wurzel’s famous paper, ‘How to Foo Bars’. Dr Drofnats has this paper in his personal .bib file as ‘mw:foo’. Dr Frobwangler has it in her personal .bib file as ‘wurzel:bars’. Each cites accordingly.

One possibility of course is to produce a unified .bib file for the whole document, and edit the citations in each paper. And sometimes that might be necessary. But if the bibliographies are *truly* self-contained and free-standing, it should be possible to use Dr Drofnats’ file for his paper, and Dr Frobwangler’s file for hers, without the trouble and risk of error that there is if files are combined.

Identifying segments and sections

Obviously, Biblatex needs to know where each segment or section begins and ends. There are three ways you can do this:

1. Most explicitly using `\begin{refsection} ... \end{refsection}` or `\begin{refsegment} ... \end{refsegment}`.
2. Using the commands `\newrefsection` and `\newrefsegment`. Each of these ends the existing refsegment or refsection (if need be) and starts a new one. You can mark the end of refsegments or refsections with `\endrefsegment` or `\endrefsection`.
3. By telling Biblatex automatically to start a new refsegment or refsection whenever a new part, or chapter, or section begins. To do this you pass the option `part`, `chapter`, `section` or `subsection` as a value for the `refsection` or `refsegment` option when loading Biblatex. So to make each chapter a free-standing section, you would put

```
\usepackage[...
  refsection=chapter]{biblatex}
```

```
\begin{refsection}
[section 1]
\end{refsection}
\begin{refsection}
[section 2]
\end{refsection}
```

```
\newrefsection
[section 1]
\newrefsection
[section 2]
\endrefsection
```

```
\chapter{One}
[section 1]
\chapter{Two}
[section 2]
\endrefsection
```

Identifying resources for refsections

Refsegments share the bibliography resources (.bib files) of the entire document. Refsections can have their own resources. The full details are rather complex. What follows is a simplified set of instructions.

1. If you want one bibliography database or databases — whether that be undivided, or divided into segments, or divided into sections but with the same .bib files used in all sections — then just use `\addbibresource{}` in the preamble.
2. If you are using refsections and you want each to have its *own* .bib file(s), then proceed as follows:

- In the preamble, use `\addbibresource{}` to identify each resource, but make use of the optional label argument to give each a label:

```
\addbibresource[label=drofnats]{./bibfiles/drofnats.bib}
\addbibresource[label=frobwangler]{./bibfiles/frobwangler.bib}
```

- Use explicit refsection commands.¹ As each section begins, include a list of the labels of the applicable resources as an optional argument:

¹ Either `\begin{refsection}` and `\end{refsection}` or `\newrefsection`.

```
\newrefsection[drofnats]
...
\newrefsection[frobwangler]
```

- If there are .bib files that you want to have available in all sections then either identify them as shown above

```
\addbibresource[label=everyone]{./bibfiles/common.bib}
...
\newrefsection[drofnats,everyone]
```

or, instead of `\addbibresource{<filename>}` use `\addglobalbib{<filename>}` in the preamble: this will declare a resource which is automatically made available to every refsection. So

```
\addglobalbib{./bibfiles/common.bib}
```

will make `common.bib` accessible to every refsection, without any need to include it in the optional argument.

Printing the bibliographies

To print a bibliography for a single refsection, you do one of two things. First, you can explicitly identify the section for which you want the bibliography printed. You do this using the section's *number*:

```
\printbibliography[section=1]
```

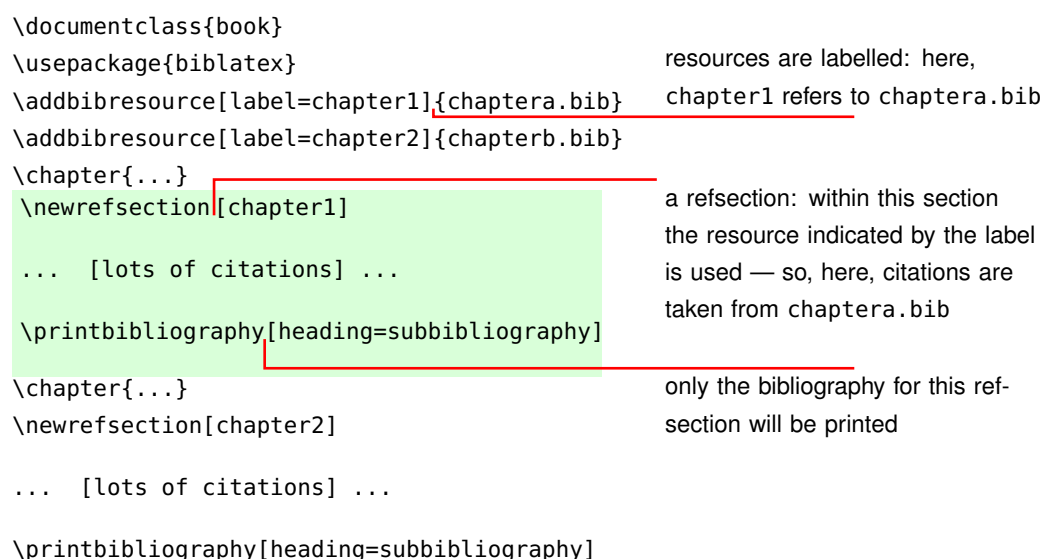
What is this number? Basically, all the sections you create (with `\begin{refsection}... \end{refsection}`, or `\newrefsection`, or implicitly at the start of chapters and so on) are numbered, starting

with 1. (Anything that happens *before* the first section or *after* the last is regarded as belonging in a section numbered 0.) So you can specify the section whose bibliography you want printed.

You can also do this for refsegments too in just the same way: the first segment you create is refsegment 1, the next is 2 and so on – and you can choose which gets printed:

```
\printbibliography[segment=2]
```

With *refsections only* (not with segments) there is another way. If you include a `\printbibliography` command *within* a refsection, Biblatex will assume that you only want to include that section.



```

\documentclass{book}
\usepackage{biblatex}
\addbibresource[label=chapter1]{chaptera.bib}
\addbibresource[label=chapter2]{chapterb.bib}
\chapter{...}
\newrefsection[chapter1]
... [lots of citations] ...
\printbibliography[heading=subbibliography]
\chapter{...}
\newrefsection[chapter2]
... [lots of citations] ...
\printbibliography[heading=subbibliography]

```

resources are labelled: here, chapter1 refers to chaptera.bib

a refsection: within this section the resource indicated by the label is used — so, here, citations are taken from chaptera.bib

only the bibliography for this refsection will be printed

Figure 7.1: Exemplary refsections

Finally, because a possible use case is to loop through each section or segment, printing the bibliography for that section or segment, and then doing the same for the next, there are special commands designed to achieve this: `\bibbysection` and `\bibbysegment`. They are ‘smart’ in that they will not print a bibliography if a particular segment or section contains no citations at all.

If you want *both* subdivided bibliographies and a main, consolidated, bibliography, there are two approaches. (1) You can use refsegments. If you then print a bibliography with *no* `segment=...` selector, all segments will be printed. (2) If *only* your cited sources are in the .bib file, you can use a `\nocite{*}` instruction within a section to print everything. Unfortunately there is no way to tell Biblatex to print a combined bibliography of multiple sections.

Some advice

In many cases, excessively divided bibliographies are more harmful than helpful. Before deciding to divide bibliographies at all, do think seriously about how a reader uses a bibliography. Don't introduce unnecessary complexity. Apart from the most obvious case (self-contained papers which need their own bibliographies, for instance) subdivision can easily be over done.

8

Sorting

A bibliography style will usually define a sorting scheme which is appropriate for that style. For instance, consider the following works:

Smith, J. (2000) *Zoology for the Amateur*. Oxbridge: Pubco.

Smith, J. (1999) *Professional Zoology*. Yarvard: Aldus.

Smith, J. (2010) *Amateur Zoology*. Camford: Otherco.

In an author/year system, it would make sense to put the works in that order. On the other hand, in an author/title system, it would probably make sense to list them as

Smith, J. *Amateur Zoology*. Camford: Otherco 2010.

Smith, J. *Professional Zoology*. Yarvard: Aldus 1999.

Smith, J. *Zoology for the Amateur*. Oxbridge: Pubco 2000.

In an alphanumeric system, the correct order will normally be one that make sense of the labels.

[Sm100] Smith J. *Zoology for the Amateur* ...

[Smi10] Smith J. *Amateur Zoology* ...

[Smi99] Smith J. *Professional Zoology* ...

A numeric system may either sort its bibliography in some way (usually along the lines of the author/title system) or print the bibliography in the order of citation in the text – a system that is conveniently and inaccurately described as involving an *unsorted* bibliography, and which one achieves by loading Biblatex with the option `sorting=none`.

Biblatex allows for a wide variety of sorting schemes, and there is considerable flexibility to define new schemes. But that largely lies outside the scope of this book, which is aimed at the ordinary user. So what we are going to do is to describe the common schemes, explain how you can make simple and reasonable changes to them, and pay a bit of attention to some special fields in the `.bib` file that can be used to influence sorting.

The built-in schemes

The following are the basic schemes, and a basic definition of what they are intended to achieve. (Note that individual bibliography styles may produce their own schemes.)

- *Unsorted*. (sorting=none) — a misnomer, really, in so far as it suggests that the bibliography might appear in some sort of random order. The bibliography will be printed in the order the works are first cited in the text.
- *Name/Title/Year* (sorting=nty). Works are first sorted by the name of the author (or editor), so all works by Albert Aardvark appear before anything written by Benjamin Badger. Within each name, the works are sorted by title. And if titles are identical, then the earlier in time is placed first.
- *Name/Year/Title* (sorting=nyt or sorting=nyvt). Works are first sorted by the name of the author (or editor), so that all works by Albert Aardvark appear before anything written by Benjamin Badger. Within each name, the works are sorted first by year and then, if there is more than one work in any given year, alphabetically by title. Name/Year/Volume/Title (sorting=nyvt) is the same as Name/Year/Title, except that the volume is considered before title.
- *Year/Name/Title* (sorting=ynt or sorting=ydnt). Sorts by the year first, then the name, then the title (can be useful for producing a chronologically organized bibliography). The difference between ynt and ydnt is that ynt works upwards (2013 comes after 2000), whereas ydnt works downwards (2000 comes after 2013).
- *Alphabetic Label/Name/Year/Title* (sorting=anyt). Sorts principally by the alphabetic label — and obviously therefore intended only for alphabetic styles. The Biblatex manual says that it then sorts by name, year and title: but so long as the labels are unique, as would usually be the case, these will never need to be consulted. There is also a style anyvt which considers volume information.
- *By order in the .bib file* (sorting=debug) this order citations by their key, and is (as its name suggests) exclusively intended for styles which are used for debugging .bib files.

Ad hoc manipulations in the .bib file.

Most of the time, Biblatex and biber will sort quite well, but there are occasions when you may need to intervene.

► Manual §§ 3.1.2.1, 3.5

Aardvark, A.	My life.	1999.
Badger, B.	Memoirs.	2005.
Badger, B.	Memoirs (2nd ed).	2010.
Badger, B.	Memories.	1999.
Badger, B.	Recollection.	2005.

Figure 8.1: nty sorting

Aardvark, A.	(1999)	My life.
Badger, B.	(1999)	Memories.
Badger, B.	(2005)	Memoirs.
Badger, B.	(2005)	Recollection.
Badger, B.	(2010)	Memoirs (2nd ed)

Figure 8.2: nyt sorting

1999.	Aardvark, A.	My Life.
1999.	Badger, B.	Memories.
2001.	Badger, B.	Memoirs.
2002.	Badger, B.	Recollection.
2003.	Badger, B.	Memoirs (2nd ed).

Figure 8.3: ynt sorting

[Aar00]	A. Aardvark.	My Life.	2000.
[Bad05a]	B. Badger.	Memoirs.	2005.
[Bad05b]	B. Badger.	Recollection.	2005.
[Bad10]	B. Badger.	Memoirs.	2010.
[Bad99]	B. Badger.	Memories.	1999.

Figure 8.4: anyt sorting

Helping out the sorting The first, and most common, is when for some reason the name field that should be printed is inappropriate for sorting, and you need to specify a slightly different version for sorting purposes only. This can happen for two main reasons.

You may just want a different name. For instance, suppose you have an ‘institutional’ author:

```
author = {{The Magoo Trust}}
```

Biblatex is going to try to sort this under T for ‘The’ — but you might think it better to have it sorted under M. In such a case, you can specify a `sortname`

```
sortname = {{Magoo Trust The}}
```

Similar things can happen with titles (indeed, it’s more common there)

```
title      = {The General Principles of EC Law}
sorttitle  = {General Principles of EC Law The}
```

If you have used a \LaTeX command in a field this may confuse the sorting, and you can use an ‘unvarnished’ version for sorting

```
title      = {\TeX\}ing,
sorttitle  = {Texing}
```

You can set `sortname`, `sorttitle`, and `sortyear` fields for these purposes.

Fiddling with the order The second change you might sometimes want to make is something which more drastically manipulates the order.

Take, for instance a book with no author or one with no date. In a name/title/year system, the authorless book gravitates to the top of the list, and in a year/name/title system, the dateless work ends up at the end of the list. Suppose you want the reverse?

The trick is this. Every entry in your `.bib` file is assumed to have a field called `presort` magically set to `mm`. And, as the name suggests, that is the first field that gets sorted. So if you set `presort` to something higher in the alphabet than `mm` (like, say, `aa`) the work in question will magically appear (alongside everything else with the same `presort` code) above the rest of the list; and if you set it to something lower in the alphabet than `mm` (like, say, `zz`) it will drop to the bottom.

The `presort` field could be used — and in the past sometimes was — for other purposes, and in particular for producing topic-based bibliographies (for instance by giving all primary sources a `presort` of

aa, to move them to the top of the list). With Biblatex there are better ways of achieving that sort of result.

In desperation you can also use `sortkey` to fix absolutely and unequivocally the ‘key’ by which a work will be sorted. Generally speaking, though, this is a counsel of desperation, and it is hard to think of a real-life situation in which it would be advised.

9

Languages

Dealing with different languages is not straightforward; but Biblatex is fairly well set up for the task. This section aims to provide basic information; there is actually considerably more on offer in Biblatex for sophisticated language use than this section deals with. This section basically assumes the ‘normal’ user who produces documents in a single language, but may occasionally need to deal with bibliographic sources in other languages.

There are a number of different ways that a bibliographic system may need to adjust to different languages. Most basically, a number of words and abbreviations may need to change: ‘ed’ in English becomes ‘éd’ in French, or ‘Hrsg’ in German. There may be differences in punctuation (for instance in how quotations are typeset). There may be differences in sorting. (For instance, is ‘Aardvark’ a word you would expect to appear at the beginning or the end of a dictionary? An English person and a Swede would disagree!) At a still more basic level – unseen by the user until it goes wrong – the hyphenation of different languages varies. And, even further ‘under the skin’ there are issues about the ‘encoding’ of an input file: how human-readable glyphs are turned into machine-readable sequences of numbers.

A dogmatic approach

Later sections of this chapter try to provide a better explanation of how all these various things can be done. But here is a dogmatic ‘just do this’ approach, which will work in many cases.

1. If you are writing in a language which requires accents and diacritics, prefer to use UTF-8 encoding in your source and .bib files. Set your editor to use that encoding and, unless you are using XeTeX or LuaTeX (which handle unicode natively), use the `inputenc` package in your source file.

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

2. Load babel, or, if you use it, polyglossia, with your preferred language.

```
\usepackage[⟨language⟩]{babel}
```

This should be done, and any language selection made, *before* loading Biblatex.

3. Load csquotes with an appropriate quotation style:

```
\usepackage[style=⟨language⟩]{csquotes}
```

4. Load Biblatex with the style autolang=hyphen, which will make sure that appropriate hyphenation patterns are used based on the langid field of the entry, if it has been set. Set that field where you think it will be helpful.

5. If you are using anything other than the standard bibliography style, consult that style's documentation to find out whether you need to 'connect' your language to a particular style-specific language style. If so, do that using

```
\DeclareLanguageMapping{⟨language⟩}{⟨language-definition⟩}
```

For instance, in the APA style, I might have

```
\DeclareLanguageMapping{british}{english-apa}
```

The language options are: american (English), australian (English), austrian, brazil, british (English), canadian (English), canadien (French), catalan, czech, danish, dutch, finnish, french, german, greek, italian, naustrian, ngerman, norsk, nynorsk, portugues, russian, spanish, swedish.

Some more detail

Encodings

Biblatex (or, more accurately, Biber) is perfectly happy working with unicode (indeed, internally this is what it always tries to do). It is generally best, unless your use of accents and diacritics is minimal, to use a UTF-8 encoded .bib file. This need not be the encoding used in your .tex file: the software will attempt to detect that, and will output a .bbl file accordingly. But in general it makes sense, nowadays, to use a UTF-8 encoded .tex source. For this, you should either use XeTeX or LuaTeX, which handle unicode natively, or use the inputenc package to set the encoding to utf8.

Occasionally, notwithstanding such precautions, there can be difficulties caused by the way T_EX and Biber deal with encodings. In such cases it may be necessary to set input and output encodings explicitly; on this, you should consult the Biber documentation.

Language

In general, in the \LaTeX world, language features are controlled using the (older) `babel` or (newer) `polyglossia` packages to set languages. The `Biblatex` package generally tries to leverage these packages to work out what language is being used. It then attempts to load a file which contains suitable bibliographical features in that language.

This is in many cases enough. But internally the position has to be a bit more complex, because `Biblatex` has to associate a given basic language with a set of files which define suitable abbreviations, formatting of dates, and so forth for that language. In the standard styles, this is entirely automatic. In non-standard styles, there may be cases where the style is language-specific, or in which you are otherwise required to tell `Biblatex` explicitly what language definitions to use.

This is the purpose of the `\DeclareLanguageMapping` command. What it does is tell `Biblatex` what definitions to use *for a particular language*. So, for instance

```
\DeclareLanguageMapping{british}{british-apa}
```

tells `Biblatex` that if the language being used for typesetting is `british-English`, then it should look for the relevant definitions in the `british-apa` language definition. Sometimes this has been done for you by the writer of the bibliography style; sometimes it needs to be explicitly done (especially if the style provides a limited range of language definitions).

It is perfectly permissible to have a document in more than one language: the citation and bibliography commands will use whichever language is active at the relevant time. So, for instance (to take a thoroughly feeble example) suppose we had the following:

```
...
\usepackage[french,british]{babel}
\usepackage[style=numeric]{biblatex}
\addbibresource{biblatex-examples.bib}

\begin{document}
We make reference to Aristotle \cite{aristotle:anima}.

\printbibliography

\selectlanguage{french}

\printbibliography
...
```

We make reference to Aristotle [`aristotle:anima`].

Figure 9.1: A document using two languages

That code will produce something which looks like figure 9.1. The first time the bibliography is printed, English conventions are used, because English is the active language. Then French is selected as the language, and this time the bibliography, when printed, is formatting according to French conventions and terminology.

In that example, the language is being selected explicitly in the source file. But what about the `.bib` entries themselves? Can they select languages? The short answer is, Yes. But it's really quite important to give a longer answer.

A `.bib` entry may have the field `language` set. But this will not affect the language conventions used to typeset it. The purpose of the `language` and `origlanguage` fields is to record *bibliographical* data, which may (depending on the style) be printed.¹ So `language` and `origlanguage` relate to the *source* not the *entry*. However there is a field `langid`² which is specifically designed for this purpose. You set the `langid` field to indicate that the language of the entry should (potentially) be fixed in a particular way. For instance, if you are citing an author and a title in a foreign language, it may be natural to specify an appropriate language ID:

¹ For instance, a style might print: Aristotle, *De Anima* (Greek).

² `langid` was previously called `hyphenation`.

```
@book{swann,
  author      = {Proust, Marcel},
  maintitle   = {À la recherche du temps perdu},
  volume      = 1,
  title       = {Du côté de chez Swann},
  date        = {1913},
  langid      = {french},
}
```

However, simply specifying the language ID does not directly make any difference: the entry simply gets printed in the ordinary way (see figure 9.2).

References

Proust, Marcel. *À la recherche du temps perdu*. Vol. 1: *Du côté de chez Swann*. 1913.

Figure 9.2: The language ID: not necessarily any difference

However, you can make it make a difference by using an option when Biblatex is loaded:

- `autoLANG=hyphen` will activate hyphenation rules for the specified language. This may or may not make any difference (depending

on line breaks and so forth). In our example it would not. But it's generally a good idea to allow hyphenation of titles and so forth correctly.

- `autolang=other` will apply the *full* set of rules for the language in question when printing that particular entry, as can be seen in figure 9.3. Note in this case that the bibliography in general is in English format (the title is 'References' not 'Références', and so forth), but the particular entry takes a French format.

Figure 9.3: `autolang=other`

In general terms, it seems to me, activating language-specific hyphenation is normally a sensible idea, but activating full language support seems to me generally a rather peculiar thing to do in general (though there might be some multilingual works where it makes sense).

There are further and deeper complexities for users of polyglossia, for whom still more complex features are available. But those are beyond the scope of this book.

Sorting

There is a tendency to consider alphabetical order, drilled into us as children, as if it were fixed and immutable. But in fact alphabetical order can vary from language to language. For instance, in Czech and Slovak accented letters are conventionally sorted after unaccented letters (so *Ěmil* comes after *Emily*), in Norwegian 'Aa' is considered the same as Å, and placed at the end of the alphabet (so *Aardvark* comes after *Zebra*!).

This sort of detail is under the control of Biber, not of Biblatex. Normally you can assume that an appropriate locale will be set based on environment variables in your computer. But you may sometimes need to specify the locale. You can do this by specifying a `sortlocale` as an option when loading Biblatex. So, for instance, to have a Norwegian sorting scheme, you would specify

```
\usepackage[sortlocale=nb_NO]{biblatex}
```

You should³ see the effect of this in figures 9.4 and 9.5. The bibliography used is the same in each case, but figure 9.4 is sorted using `sortlocale=en_GB`, whereas figure 9.5 is sorted using `sortlocale=nb_NO`.

³ Though at the time of writing Biber seems to be broken.

Aardvark, A. (2010). <i>Coming First</i> . Wildlife Press. Zebra, Z. (2005). <i>Last But Not Least</i> . Savannah UP.
--

Figure 9.4: Sorted with
sortlocale=en_GB

Zebra, Z. (2005). <i>Last But Not Least</i> . Savannah UP. Aardvark, A. (2010). <i>Coming First</i> . Wildlife Press.
--

Figure 9.5: Sorted with
sortlocale=nb_NO

Limitations

Although Biblatex has many features designed to deal with different languages, there are still limits. The limits are, as things stand, reached when different languages have fundamentally different ideas about the sort of information that should be output, or the order in which it should be output. To cope with this it is necessary to carry out deep changes to the bibliography drivers, and the existing styles do not support this.

10

Recipes

A comprehensive guide to customization is outside the scope of this book. But some requests for customization are very common, and it seems worthwhile to have a short chapter that provides some common recipes.

Names

I want at most 1/2/3/4 names printed before ‘and others’. You need to change the value of `maxbibnames`, `maxcitenames`, `minbibnames` and/or `mincitenames`. As you might expect `maxbibnames` and `minbibnames` deal with what is printed in the bibliography, and `maxcitenames` and `mincitenames` deal with what is printed in a citation (if citations print names). You can change both with `maxnames` and `minnames`, which is what you normally want to do, and what we will do here. You set this value when loading Biblatex:

```
\usepackage[maxnames=2, minnames=1]{biblatex}
```

The `maxnames` number holds the number of names that will *trigger* truncation to ‘and others’ (or ‘et al’, or whatever). The `minnames` number is the number of names *to which the list will be truncated if truncation occurs* (which must, obviously, be no more than the number that triggers truncation).

Suppose we had an entry with four authors:

```
...
author = { Alpha, A. and Beta, B.
          and Gamma, C. and Delta, D. }
...
```

The effect of various different settings of `maxnames` and `minnames` is shown in table 10.1.

Sometimes you will find that you get results which are not what you are expecting. When it truncates a name, Biblatex can be configured to consider whether the resulting list is unique. If it isn’t,

maxnames	minnames	result
1	1	A. Alpha et al
2	1	A. Alpha et al
2	2	A. Alpha, B. Beta, at al
3	1	A. Alpha et al
3	2	A. Alpha, B. Beta, et al
3	3	A. Alpha, B. Beta, C. Gamma, et al
4	any	A. Alpha, B. Beta, C. Gamma, and D. Delta

Table 10.1: Effect of maxnames and minnames

Biblatex may add extra names in order to produce a unique list. For example, suppose you are citing one work by Tom, Dick, and Harry; and one work by Tom, Dick, and Harriet, and have set maxnames to 2 and minnames to 1. Simply applying the ordinary rules, each work would end up with the authors given as ‘Tom et al.’ But in some styles it may sometimes make sense to bend the rules if by doing so it is possible to produce a ‘unique’ citation. This is done by setting the option `uniquelist=true`. In fact many standard styles (such as author/year and author/title styles *already do that*. So if you think that Biblatex is ‘disobeying’ your instructions about truncation, you might try adding the option `uniquelist=false` when loading it.

► Manual 3.6.1

I want something different from ‘et al’ printed when names are truncated. What gets printed in this case depends on the setting of two bibliography strings, and one delimiter:

- The bibstring `andothers` (by default ‘et al’)
- The delimiter `andothersdelim` (by default ‘ ’)
- The `\finalandcomma` (printed wherever a list would end in ‘and’)

So to print ‘& al.’ rather than ‘, et al’, one would put

```
\DefineBibliographyStrings{english}% or your language
{ andothers = {\& al\adddot}}
...
\renewcommand{\finalandcomma}{}
```

The redefinition of `\finalandcomma` needs to come *after* `\begin{document}`.

Things are a bit more complicated if you want to redefine `et al` in such a way that it will be different depending on how truncated the name is: for instance if you wanted ‘and others’ if more than two names were truncated, but ‘and another’ if only one was. For that purpose you would need the more extensive changes shown in figure 10.1.


```

\NewBibliographyString{andanother}
\DefineBibliographyStrings{english}% or your language
{ andothers = {and others},
  andanother = {and another}}
\renewbibmacro*{name:andothers}{%
  \ifboolexpr{
    test {\ifnumequal{\value{listcount}}{\value{liststop}}}
    and
    test \ifmorenames
  }
  {\ifnumgreater{\value{liststop}}{1}
    {\finalandcomma}
    {}}%
  \ifnumgreater{\value{listtotal} - \value{liststop}}{1}
  {\andothersdelim\bibstring{andothers}}
  {\andothersdelim\bibstring{andanother}}
}
{}}

```

Figure 10.1: Introduction of ‘and others’ and ‘and another’

I want/don't want multiple references to a single author to appear as a long dash. In the author/year, author/title and verbose styles, you can control this with the option `dashed=true` or `dashed=false` when loading Biblatex. Non-standard styles are not obliged to provide this option, but most will do so.

I want to have all names abbreviated to initials. Set the option `giveninits=true` when loading Biblatex.

I want to have initials printed in a different way. By default, the standard styles of Biblatex print initials with spaces and abbreviation dots:

P. M. Stanley

But other methods are available. If you set `terseinits=true` the initials are printed in a condensed way:

PM Stanley

If you want a ‘halfway house’, you can redefine `\bibinitperiod`. By combining `terseinits` with `\renewcommand{\bibinitperiod}{\adddot}` you can produce

P.M. Stanley

While by combining `terseinits=false` with `\renewcommand{\bibinitperiod}{}` you can produce

P M Stanley

There is, however, a catch. None of these changes will operate unless Biblalex is producing the initials. In other words, none works unless the `giveninits` option is in effect. Where names are being printed in full, initials that you have entered will be printed (as entered) with full stops and spaces, regardless of the settings of `terseinits` or `\bibinitperiod`. (There are ways around this, but they are beyond the scope of this section: the curious might look at the `OSCOLA` package code, which does something along these lines.)

I want names printed in bold/italic/small capitals etc. How names are printed depends on four standard commands: `\mkbibnamegiven` (which prints the initials or first names), `\mkbibnameprefix` (which prints the ‘de’ or ‘von’ parts), `\mkbibnamefamily` (which prints the last name) and `\mkbibnamesuffix` (which prints the ‘Jr’ part). Each or all of these can be redefined to alter the way names are printed. Each takes a single argument (the relevant part of the name), and can be used to format it.¹

Suppose, for instance, that we had the following (peculiar) requirements: We want first names printed in bold, we want ‘von’ and ‘Jr’ parts printed emphatically, and we want the last name in small capitals. We define the following

```
\renewcommand{\mkbibnamegiven}[1]{\mkbibbold{#1}}
\renewcommand{\mkbibnameprefix}[1]{\mkbibbemph{#1}}
\renewcommand{\mkbibnamesuffix}[1]{\mkbibbemph{#1}}
\renewcommand{\mkbibnamefamily}[1]{\textsc{#1}}
```

And obtain results as shown in figure 10.2.

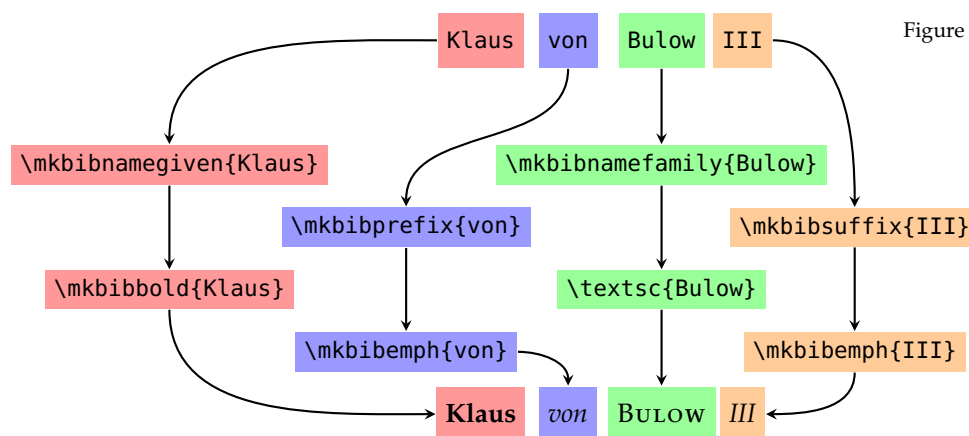


Figure 10.2: How names are styled

Of course, it's usually unnecessary to have so many different definitions. A much more common approach would be to have all the

¹ For formatting emphatically or in bold, you should use `\mkbibbold` or `\mkbibbemph`

various parts of a name formatted in the same way. But the flexibility is there if you need it.

► Biblatex Manual § 12

I want the names to be printed ‘Firstname Lastname’, but I’m getting ‘Lastname, Firstname’ Although there is, in theory, infinite flexibility, in practice there are three common patterns to how names are printed:

- In ‘ordinary’ order: given name then family name — Joe Bloggs and John Doe.
- In ‘reverse’ order: family name then given name — Bloggs, Joe and Doe, John.
- With the given name in the list reversed, but the other in ordinary order — Bloggs, Joe and John Doe.

Styles generally set these sensibly, but if you do not like the choice, you can usually change it without much difficulty. The command you need is

```
\DeclareNameAlias{sortname}{order}
```

Where *order* can be given-family (for ordinary order), family-given (for reversed order) or family-given/given-family (for the hybrid form). Some styles reset this at the start of the bibliography, in which case you *also* need (in your preamble)

```
\AtBeginBibliography{\DeclareNameAlias{...}}
```

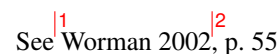
I want to produce a publication list of my own papers. People writing CVs often want to produce a bibliography of their own papers, but modifying the name format so that instead of placing their name wherever it might happen to be in the list (it could be with ‘and others’!) the list reads ‘with ...’. For this purpose you can use the biblatex-publist style, which is specifically designed for it. You tell the package your name (using the `\omitname` macro (or package option).

Punctuation issues

There are many configurable punctuation marks in Biblatex—too many to deal with comprehensively here. In this section I’m only going to cover some particularly common cases.

I want different punctuation before a postnote. The overall punctuation of a citation is shown in figure 10.3. The point marked (1) is the place between the pre-note and the beginning of the citation where Biblatex will insert `\prenotedelim`. Here that command is defined to print

In general, I use the technique of redefining macros. An alternative method, using `\DeclareDelimFormat` is also possible, and may be better if you are writing a complex style. For an example see p 45.



See Worman 2002, p. 55

Figure 10.3: Citation punctuation

nothing, so in fact nothing gets inserted. The point marked (2) is the place between the citation and the post-note where Biblalex will insert the `\postnotedelim`. In this case that is a spaced comma.

So if we redefine both these commands, we can alter what gets inserted:

```
\renewcommand{\prenotedelim}{\addcolon\space}
\renewcommand{\postnotedelim}{\space--\space}
```

I am using an author/year style, and I want a colon/a comma/nothing between the author and the year. This is controlled by the `\nameyear delim` macro (see figure 10.5). In the standard styles it is, as you can see, a space. If we wanted to make it, say, a comma, we could do so as follows:

```
\renewcommand{\nameyear delim}{\addcomma\space}
```

to produce the revised punctuation that is marked in figure 10.6.

I have multiple citations and I want different punctuation between them. As can be seen from figure 10.7

the mark placed between most multiple citations is defined either by `\multicitedelim` or by `\bibrangedash`. You are very unlikely to want to redefine the latter command (which is used in many other places), but `\multicitedelim` can easily be redefined (a comma and a space, or a semicolon and a space being the most common). To give a (silly) example

```
\renewcommand{\multicitedelim}{\slash}
```

would give us the rather bizarre results show in figure 10.8.

Between `SUPERSCRIPT CITATIONS` the delimiter is not `\multicitedelim`, but `\supercitedelim`, so if you use superscript citations it is this that you will have to redefine.

I don't want punctuation after 'In'. Standard Biblalex styles print a colon after 'In.' This is produced by `\intitlepunct`, as shown in figure 10.9.

Gaonkar, Dilip Parameshwar (2001). "On Alternative Modernities". In: *Alternative Modernities*. Ed. by Dilip Parameshwar Gaonkar. Durham and London: Duke University Press, pp. 1–23. ISBN: 0-822-32714-7.

You can therefore redefine `\intitlepunct`, for instance to

See: ¹Worman 2002² – p. 55

Figure 10.4: Citation punctuation revised

Worman¹2002

Figure 10.5: `\nameyear delim`

Worman,¹2002

Figure 10.6: `\nameyear delim` revised

[¹1, 4] [²1–3]

Figure 10.7: Multiple citations

[¹1/4] [¹1/2/¹3]

Figure 10.8: Multiple citations redefined

Figure 10.9: In: → In

```
\renewcommand{\intitlepunct}{\addspace\nopunct}
```

to avoid any punctuation being added at that point.

I want to remove the reference to ‘in’ altogether. In that case

```
\renewbibmacro{in:}{}%
```

The trouble with this is that it removes ‘in’ not only from, say, articles, but also from other types of source (such as incollection) where it may still be needed. In that case, you need a more complex revision.², using `\ifentrytype{<type>}`. The following, for instance will suppress ‘in’ for article and inproceedings only, and you should be able to work out how to adapt it.

```
\renewbibmacro{in:}{%
  \ifboolexpr{%
    test {\ifentrytype{article}}%
    or
    test {\ifentrytype{inproceedings}}%
  }
  {}%<-nothing
  {\printtext{\bibstring{in}\intitlepunct}}%<-normal ‘in’ with punctuation
}
```

I am using a numeric style, but I want my citations to appear in parentheses (1) instead of brackets [1]. This is a surprisingly difficult change to make. The code required is as follows:³

```
\makeatletter
```

```
\newrobustcmd*{\parentxttrack}[1]{%
  \begingroup
  \blx@blxinit
  \blx@setsfcodes
  \blx@bibopenparen#1\blx@bibcloseparen%
  \endgroup}
```

```
\AtEveryCite{%
  \let\parentxt=\parentxttrack%
  \let\bibopenparen=\bibopenbracket
  \let\bibcloseparen=\bibclosebracket}
```

```
\makeatother
```

```
\DeclareFieldFormat{labelnumberwidth}{\mkbibparens{#1}}
\DeclareFieldFormat{labelformat}{\mkbibbrackets{#1}}
```

² Thanks to Herbert: <http://tex.stackexchange.com/questions/10682/suppress-in-biblatex>

```
\cite: (1)
\textcite: Gaonkar (1)
```

Figure 10.10: Brackets replaced by parentheses

³ Using the clever method devised by Audrey Boruvka: <http://tex.stackexchange.com/a/16792/5404>.

I am using an author/year style: I want to replace round parentheses with square brackets: [Author 1978] instead of (Author 1978). This is more-or-less the opposite problem to the last one, and again it takes a surprising amount of work to get his right. The relevant code is as follows (to go in the preamble).

```
\makeatletter

\newrobustcmd*{\parentxttrack}[1]{%
  \begingroup
  \blx@blxinit
  \blx@setsfcodes
  \blx@bibopenbracket#1\blx@bibclosebracket%
  \endgroup}

\AtEveryCite{%
  \let\parentext=\parentxttrack%
  \let\bibopenbracket=\bibopenparen%
  \let\bibclosebracket=\bibcloseparen}

\makeatother
```

If you also want the year references to be printed in square brackets in the bibliography, you need this in addition:

```
\renewbibmacro*{date+extrayear}{%
  \iffieldundef{\thefield{datelabelsource}year}
  {}
  {\printtext[brackets]{%
    \iffieldsequal{year}{\thefield{datelabelsource}year}
    {\printdateextralabel}%
    {\printfield{labelyear}%
      \printfield{extrayear}}}}}%
\renewbibmacro*{date}{}%
\renewbibmacro*{issue+date}{%
  \iffieldundef{issue}
  {}
  {\printtext[brackets]{\printfield{issue}}}%
  \newunit}
```

I would like the parts of a bibliographical entry to be separated by commas, but the standard styles use full stops. The division between the main parts of an entry are marked by \newunitpunct. As you can see from figure 10.11, it is quite frequently used.

If we modify it by

Gaonkar, Dilip Parameshwar. “On Alternative Modernities”. In: *Alternative Modernities*. Ed. by Dilip Parameshwar Gaonkar. Durham and London: Duke University Press, 2001, pp. 1–23. ISBN: 0-822-32714-7.

Figure 10.11: The \newunitpunct punctuation

`\renewcommand{\newunitpunct}{\addcomma\space}`

we get a less choppy result: figure 10.12. Notice that the capitalization adjust automatically. It may, however, sometimes be necessary to

Gaonkar, Dilip Parameshwar, “On Alternative Modernities”, in: *Alternative Modernities*, ed. by Dilip Parameshwar Gaonkar, Durham and London: Duke University Press, 2001, pp. 1–23, ISBN: 0-822-32714-7.

Figure 10.12: The \newunitpunct punctuation

adjust other punctuation marks to ensure a really smooth result.

I am not happy with the way quotation marks are being put around article names. This is largely a function of how `csquotes` is loaded.

- Loaded with the option `style=british`, you will get single quotation marks: ‘Like This’.
- Loaded with the option `style=american`, you will get double quotation marks: “Like This”.
- If, in addition to loading `csquotes` with the option `style=american` you load `babel` or `polyglossia` with the `american` language you will get the American style of handling punctuation and quotations, where punctuation is moved inside the quotation: “like this,” in most cases.

Of course, you can equally use `csquotes` with «French», „German“, «Spanish», or any other style of quotation.

Issues about what gets printed

I don’t want to print URLs/EPRINT/DOIS. These can be controlled with the options `url`, `doi` and `eprint`. By default all of these are set `true`; to turn off any of them set the relevant option to `false`. If you want links to be created you should load the `hyperref` package.

I want to control how dates get printed. Biblatex may print a number of different dates at various points in a citation, and the format for each of them can be separately controlled, using the options given in table 10.2.

There is a bewildering range of possible settings for these various date fields. Exactly what will get printed depends may depend on the language you have set in `babel` or `polyglossia`: a date printed as ‘15th January 2000’ in `british` will be printed as ‘January 15, 2000’ in `american` and 15 janvier 2000 in `french`. However, tabel 10.3 gives examples for how a date of 2000-01-15 would get printed in British English which should be sufficient to give guidance for other languages. In addition, you set the `dateabbrev` option `true` to have dates

<code>date</code>	date of publication
<code>labeldate</code>	the ‘label’ date (in author/year styles)
<code>urldate</code>	the date a URL was ‘last visited’
<code>eventdate</code>	the date of a conference
<code>origdate</code>	an original date of publication
<code>alldates</code>	all dates

Table 10.2: The various options to set date

	<code>dateabbrev=false</code>	<code>dateabbrev=true</code>	<code>datezeros=false</code>
<code>long</code>	15th January 2000 ranges written in full: 15th January 2000– 17th January 2000	15th Jan 2000 15th Jan. 2000–17 Jan. 2000	
<code>short</code>	15/01/2000 ranges written in full: 15/01/2000– 17/01/2000		15/1/2000 15/1/2000– 17/1/2000
<code>year</code>	2000		
<code>comp</code>	15th January 2000 ranges compressed: 15th–17th January 2000	15th–17th Jan. 2000	
<code>terse</code>	as 15/01/2000 ranges compressed: 15–17/01/2000		15–17/1/2000
<code>iso8601</code>	2000-01-15 with ranges as 2000-01-15/2000-01-17		

Table 10.3: Various date settings

printed in an abbreviated form (15th Jan. 2000), and the `datezeros` option `true` to have numeric dates printed with leading zeros (2000-01-15), or `false` to have them printed without.

There’s some other information I don’t want printed. At this point we are beginning to enter the realm of serious changes, which are arguably beyond the sort of simple customization that this chapter is concerned with.

But let’s suppose, for instance, that you were absolutely happy with the standard author/year style, but just wanted not to print the publisher: just the place of publication. One way to do this is to ‘kid’ the style that you haven’t entered any publisher at all.

For this purpose we use a `sourcemap`, as follows

Gaonkar, Dilip Parameshwar. “On Alternative Modernities”. In: *Alternative Modernities*. Ed. by Dilip Parameshwar Gaonkar. Durham and London : Duke University Press, 2001, pp. 1–23. ISBN: 0-822-32714-7.

Figure 10.13: Removing the publisher using a sourcemap

Gaonkar, Dilip Parameshwar. “On Alternative Modernities”. In: *Alternative Modernities*. Ed. by Dilip Parameshwar Gaonkar. Durham and London, 2001, pp. 1–23. ISBN: 0-822-32714-7.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite=true]{
      \step[fieldset=publisher,null]
    }
  }
}
```

What this basically means is that for any .bib file, the publisher field of each entry is to be set empty (null). This will be done by biber before any data enters the system. No publisher is therefore printed, and so long as the style is capable of coping with the particular absence, all is well, as figure 10.13 shows.

Tools and Editors

Most people who use \LaTeX use a specialised editor (or a special mode or plugin for an editor) when working with \LaTeX . There are also special tools for maintaining bibliographies. These tools are not identical. That's sometimes a strength (people can have very strong views about their preferred editor!), but for newcomers it can be a definite weakness. In addition, many of the tools assume that you will be using \BibTeX – which was a fair assumption in the past, and some need ‘coaxing’ to encourage them to work easily with Biblatex.

In the shell

I think it is important, whatever editor you use, to know how to compile a file from the command line. Even the best tools sometimes fail, and it's surprising how often (especially with a complex project) a basic compilation from a shell will help diagnose problems. So every user should know how to open a shell, navigate to the directory in which the source file is found, and run

```
pdflatex1 <name>
biber <name>
pdflatex <name>
pdflatex <name>
```

¹ or xelatex or lualatex

The only trick is with $\langle name \rangle$ component: `pdflatex` wants to get to work on a file with a `.tex` extension; `Biber` wants to get to work on a file with the extension `.bcf`² (‘Old-timers’ may erroneously try to run `biber` on the `.aux` file, which is what \BibTeX would use.) But, so long as you use standard extensions, problems can be avoided by running the program in question on the ‘basename’: the filename *without any extension*. So the following are equivalent, if our source code is in a file called `myfile.tex`:

```
pdflatex myfile.tex    pdflatex myfile
```

² `bcf` stands for bibliography control file.

```

biber myfile.bcf      biber myfile
pdflatex myfile.tex   pdflatex myfile
pdflatex myfile.tex   pdflatex myfile

```

You won't *always* need all these runs. You generally only need to run Biber if you have either cited a new source or significantly changed your bibliography options (for instance the sorting). And you won't always need to run L^AT_EX twice after using biber. In any case, you get useful messages at the end of compilation, telling you what to do.

Nevertheless, for a project of any complexity, this sort of need to carry out multiple runs becomes tiresome, and you may want to look at automating it. There are a few ways you can do this. You could use a standard compiler program like `make`. There's nothing wrong with this, and if you are happy writing makefiles it's a perfectly reasonable way to go. But there are two L^AT_EX specific tools which are more likely to help you.

Latexmk

First is `latexmk`, which is a Perl script. It therefore requires Perl to be installed. Perl always will be on Linux or Mac OS X machines, but Windows users will need to download it (e.g. from strawberryperl.com). In principle, and for reasonably simple projects at least, you can substitute all the commands given above with the simple:

```
latexmk myfile
```

The script will then run L^AT_EX, and auxiliary programs such as `biber` and `makeindex`, as many times as necessary to produce a stable result (if that's possible). The `latexmk` script is 'Biber aware': it will detect if it needs to run Biber (or BIB_TE_X) and proceed accordingly. There are many options (described in the documentation).

Arara

The alternative modern approach is a program called `Arara`. This is a Java program (and therefore requires a Java command line to be installed). Instead of attempting to 'detect' what needs to be done to produce the final printable version, `Arara` enables you to place commented instructions in your source file, which essentially tell the program how to go about processing it. So, for instance, you might put the following at the start of your file:

```

% arara: pdflatex
% arara: biber
% arara: pdflatex
% arara: pdflatex

```

You then run (from a command line)

```
arara <filename>
```

The package then reads the comment lines in the file to work out what it should do. Effectively, you are writing makefile-like instructions, but in a conveniently simple form and in the file you will compile. You have to be explicit about them, but you may find this an advantage over the sort of ‘black box’ that `latexmk` offers. Again, there are all sorts of bells and whistles – but `Arara` is, as the example above shows, ‘`biber aware`’.

Texify

The `mikTeX` distribution comes with a command called `texify`, which is intended to do something along the lines of `latexmk`, but without the need to install Perl. This is *not* Biber-friendly: it assumes you will run `BIBTEX`. Although it is reputed to be possible to convert it to run Biber by setting the environment variable `BIBTEX` to `biber`, I have not had any success doing so. In my view, it’s better to install and use either `latexmk` or `Arara`.

Editors

There are many different editors available and commonly used for `LATEX`, most of which offer at least some support for the use of Biber and Biblatex. For some, such as Emacs with `AUCTEX`, support is already built in. In other cases it is necessary to add or adjust the editor’s setting in order to configure it to use Biber rather than `BIBTEX`. Given the vast number of editors, each slightly different, and the speed at which they change, it’s not practical to give detailed instructions here. Information, periodically updated, may be found at <http://tex.stackexchange.com/questions/154751/biblatex-with-biber-configuring-my-editor-to-avoid-undefined-citations>.

A practical alternative to configuring your editor to use Biber is to configure it to use `latexmk` or `Arara`, as described above.

Maintaining Databases

You can always maintain a database yourself, through a decent text editor. Some editors (such as Emacs) have a special mode for editing `BIBTEX` database files, which makes the process much smoother. But there is also software designed to do this for you. Two commonly used tools are `JabRef` and (for Mac OSX) `BibDesk`. Each provides a

graphical interface for entering bibliographic data, and useful reminders of the fields that you will need to complete.

Such packages are designed for maintaining a database file (or files) on a particular computer. Increasingly there are web- and cloud-based bibliography tools such as Mendeley and Zotero. Other web-based services (such as Google Scholar) will often also offer pre-formed `BIBTEX` records. These need to be scrutinized with care, because they are not always accurate or complete.

Support for Biblatex in these tools varies, and changes over time. You may be able to find useful up-to-date information at <http://tex.stackexchange.com/questions/23942/bibliography-tools-that-are-compatible-with-biblatex-and-biber>.

When Things Go Wrong

Things go wrong, and when they go wrong it can be quite infuriating trying to work out what has happened. This chapter aims to guide you through the possible symptoms you might see to try to help you to an answer.

In this chapter I am not going to deal with the detail of debugging changes to Biblatex styles, but with the sort of thing that goes rather obviously wrong for an ordinary user running properly written styles. The manifestation of this problem is nearly always the same: instead of citations, you have labels in square brackets in the printed brackets (as in figure 1.1; you see warnings in the L^AT_EX log file telling you

Citation 'blah' on page x undefined on input line y

and at the end of the log file you are encouraged to (re)run Biber.

A Systematic Approach

The first thing to do in this situation is, paradoxically, to start with a clean slate. When you are working on a long document, you will end up with all sorts of files in your working directory that you don't actually need, and sometimes they can make debugging harder. So start cleanly: delete all 'automatically created' files: in particular all .aux, .bcf, .blg and .bbl files.

If you use latexmk you can clean all these files away with the command `latexmk -c`.

Run L^AT_EX

Now, from a command line, run `pdflatex <filename>` (or whichever L^AT_EX you use. Don't worry much about the warnings: this is a first clean run, and you *expect* undefined citations; but if L^AT_EX is encountering errors, you obviously need to sort them out: for instance, if you have misspelled biblatex or attempted to load a non-existent bibliography style, you can't expect to produce citations. Once the

document compiles (with, as expected, warnings about undefined references), you can move on to the next stage.

Run Biber

The next task is to run Biber. This time, you should examine your output, either on the console or in the log file (which will have the extension `.blg`). If all has gone well you will see something like figure 12.1.

```
INFO - This is Biber 2.7
INFO - Logfile is 'punctcite13.blg'
INFO - Reading 'punctcite13.bcf'
INFO - Found 1 citekeys in bib section 0
INFO - Processing section 0
INFO - Looking for bibtex format file 'biblatex-examples.bib'
      for section 0
INFO - Decoding LaTeX character macros into UTF-8
INFO - Found BibTeX data source '/biblatex-examples.bib'
INFO - Overriding locale 'en-US' defaults 'variable = shifted'
      with 'variable = non-ignorable'
INFO - Overriding locale 'en-US' defaults 'normalization = NFD'
      with 'normalization = prenormalized'
INFO - Sorting list 'nty/global/' of type 'entry' with scheme 'nty'
      and locale 'en-US'
INFO - No sort tailoring available for locale 'en-US'
INFO - Writing 'punctcite13.bbl' with encoding 'ascii'
INFO - Output to punctcite13.bbl
```

Figure 12.1: Log of a successful biber run

If Biber has run successfully, you can go on to the next step. If you see any lines in the log which begin `ERROR` or `WARN` then there was some sort of problem, and you will need to do more digging.

ONCE BIBER HAS ACTUALLY RUN, even with warnings, you will find that the log provides a useful diagnostic tool. The most common errors are set out in table 12.1, together with their meanings and the action you should take. But this is not a comprehensive list. Occasionally, you may get other errors: in the worst case, the program may simply fail to run fully. This is very rare indeed. When it happens it is usually because of some occult problem in your `.bib` file.

To diagnose such an error, proceed as follows:

- Rename your existing `.bib` file.
- Gradually copy your `.bib` file entries back into an empty file (with the old name). Either copy a few entries at a time, or use the tech-

WARN - No data sources defined!	Your source file does not contain any <code>\addbibresource</code> commands.	Check your source file, and make sure it includes <code>\addbibresource</code> commands for every data source you need.
ERROR - Cannot find <i><database file></i> !	Biber was not able to find the <code>.bib</code> file you have referred to.	Make sure you have spelled the name correctly. Make sure the file is somewhere that \TeX can find it, either in a place in your \TeX -MF tree where it will be found, or in the same directory as the source file. (If you don't understand this, just play safe: put it with your source file!)
WARN - Entry <i><label></i> does not parse correctly. ERROR - BibTeX subsystem: <i><filename></i> ... syntax error: found "title", expected end of entry ("}" or ")") (skipping to next "@")	You have not written a particular entry correctly: usually (as in this case) by forgetting a comma between fields, sometimes an unescaped comment character (%) is the culprit.	Find the entry that did not parse, and correct it.
WARN - BibTeX subsystem: <i><filename></i> warning: 1 characters of junk seen at toplevel	There is something <i>outside</i> and entry, other than the beginning of a new entry. The usual reason for this is a stray comma between entries.	Check your <code>.bib</code> file around the place where the 'junk' appeared. Note that if this happens the rest of the file will fail to parse, and you may well get missing citations too.
WARN - I didn't find a database entry for <i><label></i>	That citation label is not included in your <code>.bib</code> file (or it is included after an error in that file from which recovery was not possible)	Often this is a typo (in either the source file or the database): check there is a source with that label, spelled exactly identically. The other common source of (many!) such warnings is that if there is a spoiled entry in the <code>.bib</code> file (leading to the message about 'junk' having been seen) all subsequent entries will be ignored too.

Table 12.1: Biber errors, and what they mean

nique of copying half the entries, then either removing half (if the first chunk failed) or progressively adding more.

- Each time you do this, rerun biber (you shouldn't need to re-run L^AT_EX). You should expect, and ignore, warnings about missing citations; you are simply looking to get a run that does not abort.
- In this way you should be able to identify the problem entries. Remove these.
- Retype the problematic entries by hand, checking that they are correct. Where an error sufficiently grave to prevent biber from running occurs, the problem seems usually to be with some unacceptable encoding in the file, and manually re-entering the entry will usually sort it out.

Once you have corrected any errors, re-run Biber. If you had to correct an error in your *source file* (e.g. to add or change an `\addbibresource` command), you will need to re-run (pdf)-L^AT_EX, and then Biber. If you only had to change entries in the `.bib` file, you should be OK with just running Biber.

Finally, run L^AT_EX again

Once you manage to run Biber without errors or warnings, re-run L^AT_EX. This time you should get a complete compilation, without any warnings.

There is one category of error that can sometimes occur at this point. Just occasionally you will find that when L^AT_EX is run you get complaints about problems with some unicode character. The problem occurs because biber works internally in one flavour of unicode normalisation, but L^AT_EX does not, and even (in fact, especially) when unicode is used (via `\usepackage[utf8]{inputenc}`) there can be occasional incompatibilities. If you see this sort of trouble, try running biber with the option `--output_safechars`.

A less systematic approach

Sometimes a less systematic approach is appropriate. If a document and bibliography have all been compiling correctly, but suddenly you start to see problems, particularly with one or two entries, go back and check those particular entries. If those look correct (the label is correctly spelled in the `.tex` source, the entry seems correct in the `.bib` file) it's worth carrying out a manual run of Biber just to check it is working correctly.

Problems with output

Problems with output have three possible sources. (a) It may be that your bibliography style is, in some way, not to your taste. This is not really a ‘bug’ as such: it may well be that the style is doing exactly what its author intended it to. There are, of course, various ways in which you may be able to tweak a style to your liking, some of which are discussed elsewhere in this document. But the problem is not one of debugging. (b) It may be that there actually *is* a bug in the bibliography style you have used. These things are complex, and bugs happen. If you are nervous about them, try to stick to well-established and maintained styles. Other possible things to do are to get help online¹ or to contact the style’s maintainer by email. (c) It may be a problem with your `.bib` file.

¹ For instance at `tex.stackexchange.com`

It’s only common sense to check the last point first. Among the common errors in a file which will ‘work’ are the following:

- Incorrect lists of names. It’s terribly easy to write

```
John Smith, A. U. Thor, Joe Bloggs
```

when you mean

```
John Smith and A. U. Thor and Joe Bloggs
```

- Incorrect names, particularly forgetting to put a full stop after initials (especially since it often makes no difference): `Smith, J` instead of `Smith, J.`
- Forgetting to put extra braces round institutional authors, so that `Press Complaints Commission` becomes ‘`P. C. Commission`’.
- Forgetting to put braces around a word whose capitalization should not be changed, so that `German` becomes `german`
- Inadvertently muddling similar fields, such as `pages` and `pagination`, `journaltitle` and `series`, `volume` and `part`.

Further help

The main source of help on Biblatex is its manual, which can be found either online or, on a properly installed \TeX system, by typing `texdoc biblatex` at the command line. Individual styles also have their own documentation, which is sometimes extensive. That too can be found using the `texdoc` command, adding the name of the package.

When the going gets tough, it's sometimes helpful to have an experienced person look at the problem. If you don't have such a guru on tap, you can always ask a question at <http://tex.stackexchange.com>, where the `biblatex` tag appears frequently (and the package's maintainers regularly check questions). If you decide to do that, please consider the following:

- Make a good faith effort to solve the problem yourself first, including by making sensible searches for duplicate questions.
- Tag your question `biblatex`.
- Post code which illustrates the problem.
- But *don't* just post or provide a link to a huge file. Make a new file, using a standard class (such as `article`) which shows it. Similarly, remove all packages that seem to be irrelevant to the problem. Preparing such a document may well help you to isolate it. For instance, if you can't reproduce the problem unless you add some odd package or antiquated class file, the solution may be obvious.
- Include a minimal bibliography file in the example itself. There's an easy way to do this:
 - Add `\usepackage{filecontents}` to your preamble.
 - Place the necessary bibliography entries, just as in your `.bib` file, within an environment that begins `\begin{\filecontents}{\jobname.bib}` and `\end{filecontents}`. This will cause \LaTeX to create such a file on its first run.
 - Use `\addbibresource{\jobname.bib}` to load that file.
- Be absolutely specific about what the problem is, and what a solution would look like. Add an image which shows what is wrong and what you need.
- Don't take it amiss if someone on the site manages to see that your question is a duplicate, and points that out. This is not an aggressive gesture. It's sometimes hard for anyone other than an expert even to *realise* that what look like two different problems are really just different manifestations of the same difficulty.

Quick Start Guide

Biblatex is a modern, and vastly more flexible, replacement for `BIBTEX` for those needing to automate bibliography production in `LATEX`. These pages provide a very quick ‘quick start’ guide. A rather more carefully explained guide is found in chapter 1, and of course much more information throughout the guide, and in the Biblatex manual.

Step 1: Create a `.bib` file

Containing the bibliographical data. Biblatex follows the basic structure of `BIBTEX`, but expands on it. An example is shown in figure 12.2.

► Chapter 2

```
@book{work:1,
author = {Aristotle},
title = {De Anima},
date = 1907,
editor = {Hicks, Robert Drew},
publisher = {CUP},
location = {Cambridge},
}
```

Key: a unique key chosen by you and used in `\cite` commands to refer to this source.

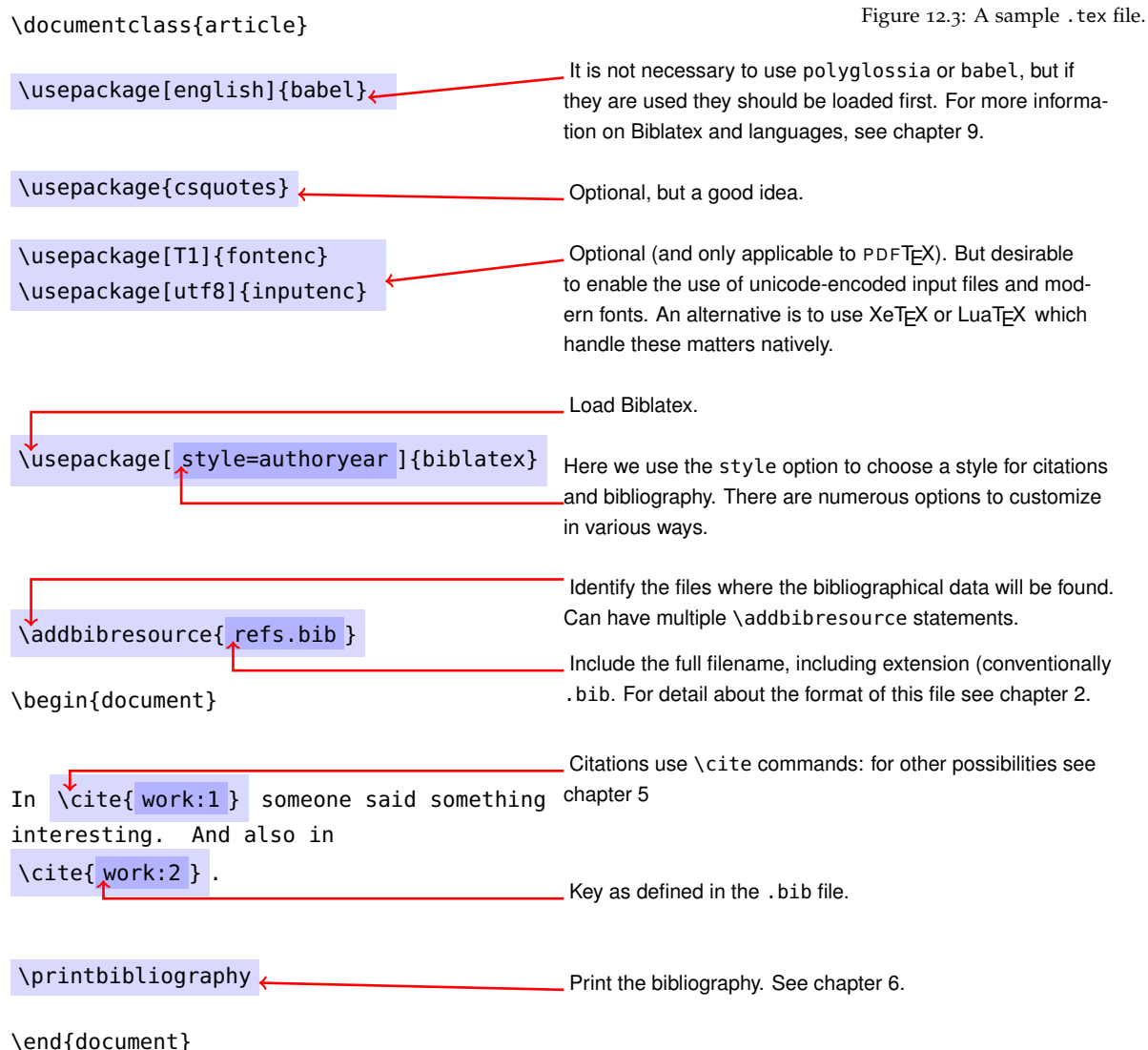
Figure 12.2: A sample `.bib` file entry

Step 2: Write source file

Prepare your source file, loading the Biblatex package, adding your `.bib` file as a resource, and including citations and a bibliography. An annotated same file appears in figure 12.3.

Step 3: Compile the `LATEX` source

Compile the `LATEX` source file, using `pdflatex`, `xelatex`, or `lualatex`.



Step 4: Run Biber

Run the Biber program to prepare bibliographical data. Do this by running

```
biber  $\langle basename \rangle$   
or  
biber  $\langle basename \rangle$ .bcf
```

Steps 5 ...: Run L^AT_EX at least once more

Run L^AT_EX at least once, and sometimes twice more. It will read in the citation data that Biber has prepared, produce the bibliography and citations, and prepare the final text.

Aide Memoire or Cheat Sheet

Load Biblatex and csquotes

```
\usepackage[backend=biber,  
style=<style>,<options>]{biblatex}  
\usepackage[style=<style>]{csquotes}
```

Always add a bibliography file with

```
\addbibliography{<bibfile.bib>}
```

Languages

```
\usepackage[<language>]{babel}  
or \usepackage{polyglossia}
```

Load these before Biblatex.

Encodings and accents Use UTF-8 in .bib file, or use T_EX accent commands \’ etc. Prefer unicode. Set input encoding of .tex file with inputenc, or use XeT_EX or LuaT_EX.

Citations Use \autocite and \cite in all styles. Use \textcite for ‘running’ citations. Use \parencite for parenthetical citations. Use \footcite for footnote citations.

Citation commands take the form:

```
\cite[<postnote>]{<key>}  
or \cite[<prenote>][<postnote>]{<key>}
```

Printing the bibliography

```
\printbibliography
```

Compilation

- Run L^AT_EX.
- Run Biber.
- Run L^AT_EX once or twice more.

Database Entries take the form

```
@<type>{<key>,
  <field> = {<value>},
  or <field> = "<value>", }
```

Common types:

- @article for journal article.
- @book for book.
- @incollection for edited collection.
- @inbook for collection by single author.

Common fields:

- title e.g. 'Bleak House'
- journaltitle e.g. 'Nature'
- author e.g. 'Shakespeare, William'
- editor e.g. 'Houseman, A. E.'
- date e.g. '2001-11-10', '1999'
- volume, issue, number
- pages e.g. '1-10'
- location e.g. 'London'
- publisher e.g. 'OUP'

Names

```
von Lastname, Firstname I.
Lastname, F. I.
Firstname Lastname
```

Multiple authors Use 'and' not ',:':

```
A. Thor and Ann Other
```

Debugging

- Did biber run?
- What error messages?
- Is bibresource correctly identified?
- Can T_EX find it?
- Are all relevant .bib entries correct?

Examples

The following pages give some simple examples. The examples are typeset as a simple article format, using Times New Roman. In each case (at least) three exemplars are cited (drawn from the `biblatex-examples.bib` database): one book, one book with an editor, one article and one incollection: I've chosen these because they seem to me to be typical of the sort of literature that an academic article will cite, and once you see these it's quite easy to predict how other entry types will be handled by the same style.

For reference, the `.bib` entries for the examples I have used are given in figures 4, 5, 6 and 7 on pages 115 to 116.

In each example I have also shown the output generated by the four main citation commands: `\cite`, `\autocite`, `\textcite` and `\footcite`. Depending on the style, not all of these commands are necessarily appropriate, of course.

```
@book{worman,  
  author      = {Worman, Nancy},  
  title       = {The Cast of Character},  
  date        = 2002,  
  publisher   = {University of Texas Press},  
  location    = {Austin},  
  langid      = {english},  
  langidopts  = {variant=american},  
  sorttitle   = {Cast of Character},  
  indextitle  = {Cast of Character, The},  
  subtitle    = {Style in Greek Literature},  
  shorttitle  = {Cast of Character},  
}
```

Figure 4: A book entry: worman

```

@book{aristotle:anima,
  author      = {Aristotle},
  title       = {De Anima},
  date        = 1907,
  editor      = {Hicks, Robert Drew},
  publisher   = cup,
  location    = {Cambridge},
  keywords    = {primary},
  langid      = {english},
  langidopts  = {variant=british},
}

```

Figure 5: A book with an editor:
aristotle:anima

```

@collection{gaonkar,
  editor      = {Gaonkar, Dilip Parameshwar},
  title       = {Alternative Modernities},
  date        = 2001,
  publisher   = {Duke University Press},
  location    = {Durham and London},
  isbn        = {0-822-32714-7},
  langid      = {english},
  langidopts  = {variant=american},
}
@InCollection{gaonkar:in,
  author      = {Gaonkar, Dilip Parameshwar},
  editor      = {Gaonkar, Dilip Parameshwar},
  title       = {On Alternative Modernities},
  date        = 2001,
  booktitle   = {Alternative Modernities},
  publisher   = {Duke University Press},
  location    = {Durham and London},
  isbn        = {0-822-32714-7},
  pages       = {1-23},
}

```

Figure 6: An incollection entry:
gaonkar:in

```

@article{reese,
  author      = {Reese, Trevor R.},
  title       = {Georgia in Anglo-Spanish Diplomacy, 1736-1739},
  journaltitle = {William and Mary Quarterly},
  date        = 1958,
  series      = 3,
  volume      = 15,
  pages       = {168-190},
  langid      = {english},
  langidopts  = {variant=american},
}

```

Figure 7: An article: reese

The Chicago Author/Date Biblatex Style

The biblatex-chicago style comes in two flavours, implementing the *Chicago Manual of Style*. The “authordate” style, shown here, is intended to put full references in the text, using an author/date system: `\autocite` is therefore equivalent to `\cite`. Thus, for instance, if we cite a book `\autocite{worman}` or an article (Reese 1958) we get the reference in the text. The `\textcite` command is important in such styles, also works as expected, thus: see Worman (2002).

There are some special features to this style. Instead of loading biblatex, you invoke it by loading the biblatex-chicago package itself (with the option `authordate` for the author/date style):

```
\usepackage[authordate,backend=biber]{biblatex-chicago}
```

The style provides support for various types of source that are not catered for by standard biblatex, such as music and recordings. It has an extensive range of options. You should consult the documentation.

Note that in this sample I have loaded `csquotes` with the `style=american` option; had I loaded it with the `style=british` we would get single quotation marks around article titles.

The bibliography here shows you straightforward examples of a book (Worman 2002), an article (Reese 1958), an edited book (Aristotle 1907) and an essay in a collection (Gaonkar 2001).

Examples of Citation Commands

```
\cite{worman}: Worman 2002
```

```
\autocite{worman}: (Worman 2002)
```

```
\textcite{worman}: Worman (2002)
```

```
\footcite{worman}: lorem ipsum dolor1
```

References

Aristotle. 1907. *De Anima*. Edited by Robert Drew Hicks. Cambridge: Cambridge University Press.

1. Worman 2002.

- Gaonkar, Dilip Parameshwar. 2001. "On Alternative Modernities." In *Alternative Modernities*, edited by Dilip Parameshwar Gaonkar, 1–23. Durham and London: Duke University Press. ISBN: 0-822-32714-7.
- Reese, Trevor R. 1958. "Georgia in Anglo-Spanish Diplomacy, 1736–1739." *William and Mary Quarterly*, 3rd ser., 15:168–190.
- Worman, Nancy. 2002. *The Cast of Character: Style in Greek Literature*. Austin: University of Texas Press.

The Chicago Notes Biblatex Style

The biblatex-chicago style comes in two flavours, implementing the *Chicago Manual of Style*. The ‘notes’ style, shown here, is intended to put full references in footnotes: `\autocite` is therefore equivalent to `\footcite`. Thus, for instance, if we cite a book¹ or an article,² we get the reference in a footnote. Repeated references to the same source, such as this,³ produce ‘Ibid’, while references to a previously cited work, like this,⁴ produce abbreviated references to author and title (but do not cross-refer back to the note where the source was first cited).

There are some special features to this style. Instead of loading biblatex, you invoke it by loading the biblatex-chicago package itself (with the option `notes` for the note style):

```
\usepackage[notes,backend=biber]{biblatex-chicago}
```

The style provides support for various types of source that are not catered for by standard biblatex, such as music and recordings. It has an extensive range of options. You should consult the documentation.

Note that in this sample I have loaded `csquotes` with the `style=british` option; had I loaded it with the `style=american` we would get double quotation marks around article titles.

The bibliography here shows you straightforward examples of a book,⁵ an article,⁶ an edited book⁷ and an essay in a collection.⁸

Examples of Citation Commands

`\cite{worman}`: Worman, *Cast of Character*

`\autocite{worman}`: ⁹

1. Nancy Worman, *The Cast of Character: Style in Greek Literature* (Austin: University of Texas Press, 2002).

2. Trevor R. Reese, ‘Georgia in Anglo-Spanish Diplomacy, 1736–1739,’ *William and Mary Quarterly*, 3rd ser., 15 (1958): 168–190.

3. Ibid.

4. Worman, *Cast of Character*.

5. Ibid.

6. Reese, ‘Georgia in Anglo-Spanish Diplomacy, 1736–1739.’

7. Aristotle, *De Anima*, ed. Robert Drew Hicks (Cambridge: Cambridge University Press, 1907).

8. Dilip Parameshwar Gaonkar, ‘On Alternative Modernities,’ in *Alternative Modernities*, ed. Dilip Parameshwar Gaonkar (Durham and London: Duke University Press, 2001), 1–23, ISBN: 0-822-32714-7.

9. Worman, *Cast of Character*.

\textcite{worman}: Worman¹⁰

\footcite{worman}: lorem ipsum dolor¹¹

References

Aristotle. *De Anima*. Edited by Robert Drew Hicks. Cambridge: Cambridge University Press, 1907.

Gaonkar, Dilip Parameshwar. ‘On Alternative Modernities.’ In *Alternative Modernities*, edited by Dilip Parameshwar Gaonkar, 1–23. Durham and London: Duke University Press, 2001. ISBN: 0-822-32714-7.

Reese, Trevor R. ‘Georgia in Anglo-Spanish Diplomacy, 1736–1739.’ *William and Mary Quarterly*, 3rd ser., 15 (1958): 168–190.

Worman, Nancy. *The Cast of Character: Style in Greek Literature*. Austin: University of Texas Press, 2002.

10. Ibid.

11. Ibid.

The OSCOLA Biblatex Style

The `oscola` style implements the style rules of the Oxford Standard for the Citation of Legal Materials. It is a footnote style in which `\autocite` produces footnoted citations.¹ Although designed for legal citations (and therefore covering a wide range of non-standard sources) it also deals with the standard types of source as well.

The package is loaded in the usual way:

```
\usepackage[style=oscola]{biblatex}
```

The bibliography here shows you straightforward examples of a book,² an article,³ an edited book⁴ and an essay in a collection.⁵ We also cite a judicial decision.⁶

Examples of Citation Commands

`\cite{worman}`: Nancy Worman, *The Cast of Character: Style in Greek Literature* (University of Texas Press 2002)

`\autocite{worman}`: ⁷

`\textcite{worman}`: Worman⁸

`\footcite{worman}`: lorem ipsum dolor⁹

References

Aristotle, *De Anima* (Hicks RD ed, Cambridge University Press 1907).

Gaonkar DP, 'On Alternative Modernities' in DP Gaonkar (ed), *Alternative Modernities* (Duke University Press 2001).

¹Nancy Worman, *The Cast of Character: Style in Greek Literature* (University of Texas Press 2002); Trevor R Reese, 'Georgia in Anglo-Spanish Diplomacy, 1736–1739' (1958) 15 *William and Mary Quarterly* 3rd series 168.

²Worman (n ??).

³Reese (n ??).

⁴Aristotle, *De Anima* (Robert Drew Hicks ed, Cambridge University Press 1907).

⁵Dilip Parameshwar Gaonkar, 'On Alternative Modernities' in Dilip Parameshwar Gaonkar (ed), *Alternative Modernities* (Duke University Press 2001).

⁶*Donoghue v Stevenson* [1932] AC 562 (HL).

⁷Worman (n ??).

⁸*ibid.*

⁹*ibid.*

Donoghue v Stevenson [1932] AC 562 (HL).

Reese TR, 'Georgia in Anglo-Spanish Diplomacy, 1736–1739' (1958) 15 William and Mary Quarterly 3rd series 168.

Worman N, *The Cast of Character: Style in Greek Literature* (University of Texas Press 2002).

The IEEE Biblatex Style

The biblatex-ieee style implements the style rules of the Institute of Electrical and Electronic Engineers. It is a numeric style, in which `\autocite` produces numbered labels thus [1]. (There is a variant which produces alphabetic labels: consult the documentation.) As can be seen below, the `\footcite{}` command makes no sense with it, and should be avoided.

The package is loaded in the usual way:

```
\usepackage[style=ieee]{biblatex}
```

I have loaded csquotes with the `lamericanl` style.

The bibliography here shows you straightforward examples of a book [1], an article [2], an edited book [3] and an essay in a collection [4].

Examples of Citation Commands

```
\cite{worman}: [1]
```

```
\autocite{worman}: [1]
```

```
\textcite{worman}: Worman [1]
```

```
\footcite{worman}: lorem ipsum dolor1
```

References

- [1] N. Worman, *The Cast of Character, Style in Greek Literature*. Austin: University of Texas Press, 2002.
- [2] T. R. Reese, “Georgia in Anglo-Spanish diplomacy, 1736–1739,” *William and Mary Quarterly*, 3rd ser., vol. 15, pp. 168–190, 1958.
- [3] Aristotle, *De Anima*, R. D. Hicks, Ed. Cambridge: Cambridge University Press, 1907.
- [4] D. P. Gaonkar, “On alternative modernities,” in *Alternative Modernities*, D. P. Gaonkar, Ed., Durham and London: Duke University Press, 2001, pp. 1–23, ISBN: 0-822-32714-7.

¹1.

The APA Biblatex Style

The biblatex-apa style implements the style rules of the American Psychological Association. It is an author/date style in which `\autocite` produces parenthetical citations (Reese, 1958; Worman, 2002), and in which `\textcite` is frequently required, thus: see Worman (2002).

The package is loaded in the usual way:

```
\usepackage[style=apa]{biblatex}
```

The one ‘trick’ with this style is that it is important to load an appropriate language mapping. So, for instance, after loading biblatex, you need to

```
\DeclareLanguageMapping{english}{english-apa}
```

(substituting, of course, your own language). If you don’t do this, you will get errors.

The bibliography here shows you straightforward examples of a book (Worman, 2002), an article (Reese, 1958), an edited book (Aristotle, 1907) and an essay in a collection (Gaonkar, 2001).

Examples of Citation Commands

```
\cite{worman}: Worman, 2002
```

```
\autocite{worman}: (Worman, 2002)
```

```
\textcite{worman}: Worman (2002)
```

```
\footcite{worman}: lorem ipsum dolor1
```

References

- Aristotle. (1907). *De anima* (R. D. Hicks, Ed.). Cambridge: Cambridge University Press.
- Gaonkar, D. P. (2001). On alternative modernities. In D. P. Gaonkar (Ed.), *Alternative modernities* (pp. 1–23). Durham: Duke University Press.
- Reese, T. R. (1958). Georgia in Anglo-Spanish diplomacy, 1736–1739. *William and Mary Quarterly*. 3rd ser., 15, 168–190.
- Worman, N. (2002). *The cast of character: Style in Greek literature*. Austin: University of Texas Press.

¹Worman, 2002.

The MLA Biblatex Style

The biblatex-mla style implements the style rules of the Modern Language Association. It is fundamentally an author/title style in which `\autocite` produces parenthetical citations (Worman). The title, however, is printed only where you cite more than one work by a single author. Citations in footnotes will produce full bibliographical information on the first occasion,¹ though this can be changed.

The package is loaded in the usual way:

```
\usepackage[style=mla]{biblatex}
```

Since MLA is an American Style, this example uses

```
\usepackage[american]{babel}
```

to ensure the correct language file is loaded; it also loads `csquotes` with the `american` option.

The bibliography here shows you straightforward examples of a book (Worman), an article (Reese), an edited book (Aristotle, *De Anima*) and an essay in a collection (Gaonkar). I have included another work by Aristotle (Aristotle, *Poetics*), in order to show how the MLA style prints titles (only) where more than one work by a single author has been cited.

Examples of Citation Commands

```
\cite{worman}: Worman
```

```
\autocite{worman}: (Worman)
```

```
\textcite{worman}: Worman
```

```
\footcite{worman}: lorem ipsum dolor2
```

Works Cited

Aristotle. *De Anima*. Edited by Robert Drew Hicks. Cambridge: Cambridge University Press, 1907. Print.

¹Aristotle, *Poetics*, edited by D. W. Lucas, Clarendon Aristotle (Oxford: Clarendon Press, 1968).

²Worman.

- . *Poetics*. Edited by D. W. Lucas. Oxford: Clarendon Press, 1968. Print. Clarendon Aristotle.
- Gaonkar, Dilip Parameshwar. “On Alternative Modernities.” *Alternative Modernities*. Edited by Dilip Parameshwar Gaonkar. Durham and London: Duke University Press, 2001. 1–23. Print.
- Reese, Trevor R. “Georgia in Anglo-Spanish Diplomacy, 1736–1739.” *William and Mary Quarterly* 3rd ser. 15 (1958): 168–190. Print.
- Worman, Nancy. *The Cast of Character: Style in Greek Literature*. Austin: University of Texas Press, 2002. Print.

The DW Author/Title Biblatex Style

The authortitle-dw style is a style produced by Dominik Waßenhoven. It was designed for use in the humanities; and it is intended to be very highly customizable. It comes in two flavours: author/title (shown here) and a verbose style. (On this occasion I have used British English conventions for quotations and so forth.)

The package is loaded in the usual way:

```
\usepackage[style=authortitle-dw]{biblatex}
```

The bibliography here shows you straightforward examples of a book,¹ an article,² an edited book³ and an essay in a collection.⁴

Examples of Citation Commands

```
\cite{worman}: Worman: Cast of Character
```

```
\autocite{worman}: 5
```

```
\textcite{worman}: Worman (ibid.)
```

```
\footcite{worman}: lorem ipsum dolor6
```

References

Aristotle: *De Anima*, ed. by Robert Drew Hicks, Cambridge 1907.

Gaonkar, Dilip Parameshwar: On Alternative Modernities, in: idem (ed.): *Alternative Modernities*, Durham and London 2001, pp. 1–23.

Reese, Trevor R.: Georgia in Anglo-Spanish Diplomacy, 1736–1739, in: William and Mary Quarterly, 3rd ser. 15 (1958), pp. 168–190.

Worman, Nancy: *The Cast of Character. Style in Greek Literature*, Austin 2002.

¹Worman: *Cast of Character*.

²Reese: Georgia in Anglo-Spanish Diplomacy, 1736–1739.

³Aristotle: *De Anima*.

⁴Gaonkar: *On Alternative Modernities*.

⁵Worman: *Cast of Character*.

⁶Ibid.

The DW Footnote Biblatex Style

The footnote-dw style is a style produced by Dominik Waßenhoven. It was designed for use in the humanities; and it is intended to be very highly customizable. It comes in two flavours: author/title and a verbose style, intended only for use in footnotes, shown here.

The style is particularly disciplinarian about requiring citations into footnotes: even `\cite` is placed in a footnote automatically (and `\textcite`, although it does give the author's name in the text, also creates a footnote reference).

The package is loaded in the usual way. For the author/title style:

```
\usepackage[style=footnote-dw]{biblatex}
```

On this occasion, I display the bibliography formatted in the German language style.

The bibliography here shows you straightforward examples of a book,¹ an article,² an edited book³ and an essay in a collection.⁴

Examples of Citation Commands

`\cite{worman}`: ⁵

`\autocite{worman}`: ⁶

`\textcite{worman}`: **worman**⁷

`\footcite{worman}`: lorem ipsum dolor⁸

¹worman.

²reese.

³aristotle:anima.

⁴gaonkar:in.

⁵worman.

⁶worman.

⁷.

⁸worman.

The Nature Biblatex Style

The biblatex-nature style implements the style rules of the Journal *Nature*. It is a numeric style, in which `\autocite` produces superscript labels thus¹. As can be seen below, the `\footcite{}` command makes no sense with it, and should be avoided: in general you should use `\autocite{}`.

The package is loaded in the usual way:

```
\usepackage[style=nature]{biblatex}
```

The bibliography here shows you straightforward examples of a book¹, an article², an edited book³ and an essay in a collection⁴.

Examples of Citation Commands

```
\cite{worman}: [1]
```

```
\autocite{worman}: 1
```

```
\textcite{worman}: Worman [1]
```

```
\footcite{worman}: lorem ipsum dolor1
```

References

1. Worman, N. *The Cast of Character. Style in Greek Literature* (University of Texas Press, Austin, 2002).
2. Reese, T. R. Georgia in Anglo-Spanish Diplomacy, 1736–1739. *William and Mary Quarterly*. 3 **15**, 168–190 (1958).
3. Aristotle. *De Anima* (ed Hicks, R. D.) (Cambridge University Press, Cambridge, 1907).
4. Gaonkar, D. P. in *Alternative Modernities* (ed Gaonkar, D. P.) 1–23 (Duke University Press, Durham and London, 2001). ISBN: 0-822-32714-7.

¹1.

Index

- `.bib` file, *see* database
- abbreviations
 - in database, *see* database
- accented letters, 19, 20
- `\addbibresource`, 74
- `\addcolon`, 34
- `\addcomma`, 34
- `\adddot`, 34
- `\addexclam`, 34
- `\addglobalbib`, 74
- `\addnbspspace`, 34
- `\addperiod`, 34
- `\addquestion`, 34
- `\addsemicolon`, 34
- `\addspace`, 34
- `\addtocategory`, 70
- Arara, 100
- article
 - database entry, *see* database
- arXiv, 26
- `\AtDataInput`, 36
- `\Ateverybibitem`, 35
- `\AtEveryCitekey`, 35, 72
- author
 - citing name, 54
 - institutional, 19
- `\autocite`, 52
- backreferences, *see*
 - bibliography, references
- `\bibbysection`, 75
- `\bibbysegment`, 75
- BibDesk, 101
- Biblatex
 - compared to bibtex, 11–13
 - package options, *see* options
 - very basic introduction, 7–9
- bibliography
 - appearance, 63–64
 - heading, 61–63
 - multiple
 - see* multiple
 - bibliographies, 67
 - printing, 61
 - references, 65
 - title, 62
- bibliography style
 - biblatex-dw, 39
- bibliography style
 - AIP, 39
 - alphanumeric
 - custom labels, 43
 - punctuation, 43
 - standard, 43
 - American Chemical Society, 39
 - Angewandte Chemie, 39
 - APA, 39
 - APS, 39
 - author/title
 - authortitle-dw, 47
 - compressing citations, 46
 - ibid, 47
 - names, 47
 - punctuation, 47

- sorting citations, 46
 - standard, 46–47
 - terse forms, 47
- author/year
 - compressing, 44
 - ibid, 44, 45
 - names, 44
 - punctuation, 45
 - standard, 44–46
- Chicago, 39
- general styles, 39
- Historische Zeitschrift, 39
- IEEE, 39
- MLA, 39
- Nature*, 39
- NEJM, 39
- numeric
 - bibliography format, 40
 - compressing citations, 40
 - punctuation, 42
 - sorting citations, 40
 - standard, 40
 - superscript, 41
- OSCOLA, 39
- philosophy, 39
- Royal Society of Chemistry,
 - 39
- SBL, 39
- Science*, 39
- specific styles, 38–39
- verbose
 - Chicago, 48
 - footnote-dw, 48
 - ibid, 48
 - standard, 48–49
- bibliography styles
 - alphanumeric
 - citation sorting, 42
 - standard, 42
 - choosing a style, 39
 - numeric
 - standard, 42
- book
 - database entry, *see* database
- capitalization
 - database, in, *see* database
- citation
 - commands
 - author, 54
 - capital forms, 54
 - entry sets, 55
 - for numerical styles, 54
 - general, 52–53
 - particular fields, 54–55
 - title, 54
 - url, 55
 - year, 54
 - general form, 51
 - general pattern, 51
 - multiple, 58–59
 - postnote, 56–58
 - customisation, 57
 - force prefixes, 56
 - Latin gadgets, 57
 - suppress prefixes, 56
 - \cite, 52
 - \citeauthor, 54
 - \citedate, 54
 - \cites, 58
 - \citetitle, 54
 - \citeurl, 55
 - \citeyear, 54
- collection
 - article in, *see* database,
 - incollection
- compiling, 99
- crossref, 29
- customisation
 - biblatex.cfg, 36
 - file locations, 36
 - general techniques, 34–36
 - hooks, 35
 - in preamble, 36
 - postnote, 57
- customization
 - bibliography appearance, 63
 - bibliography heading, *see* bibliography, heading

- bibliography title, *see*
 - bibliography, title
- citations in brackets, 93
- dashes in bibliography, 88
- date, 95
- et al*, 88
- font
 - url, 95
- 'In'
 - removal, 93
- initials, 89
- labels in parentheses, 93
- list of own papers, 91
- names, 87, 90, 91
- punctuation, *see* punctuation
- suppressing information, 96
- URL
 - font, 95
- CV
 - list of papers, 91
- dash
 - in bibliography, *see*
 - customization, dashes
- database
 - abbreviations, 31–32
 - article type, 27–29
 - book type, 22, 27
 - capitalization, 20
 - crossref, 29
 - dates, in, 20
 - electronic publication, 25
 - entry type, 15
 - fields, 18
 - general description, 15, 16
 - inbook entry, *see* database
 - incollection type, 29–30
 - ISBN, 26
 - keyword field, 70
 - `\langid` field, 84
 - languages, 21
 - L^AT_EX in, 79
 - names, 18
 - names, in, 19
 - presort field, 79
 - sortkey, 80
 - sortname field, 78
 - sorttitle field, 79
 - URL, 25
- date
 - citing date, 54
 - customization, 95
 - database, in, *see* database
- debugging
 - empty citations, 103
 - errors in .bib file, 104
 - general approach, 103–106
 - output problems, 107
- `\DeclareBibliographyCategory`, 70
- `\DeclareFieldFormat`, 35
- `\DeclareListFormat`, 35
- `\DeclareNameFormat`, 35
- `\defbibcheck`, 72
- `\defbibentryset`, 55
- `\defbibenvironment`, 63
- `\defbibfilter`, 71
- `\defbibheading`, 63
- `\DefineBibliographyStrings`, 34
- editors, 101
- encoding, 82
- entryset, 55
 - in .bib file, 55
 - using `\defbibentryset`, 55
- errors, *see* debugging
- `\ExecuteBibliographyOptions`, 36
- files
 - `biblatex.cfg`, 36
- formatting
 - field formats, 35
- ibid
 - author/title styles, 47
 - author/year styles, in, 44

- ibidtracker option, 45
 - in verbose styles, 48
- indexing, 65
- institutions
 - as authors, 19
- ISBN
 - database, in, *see* database
- JabRef, 101
- JSTOR, 26
- language
 - specification in database, *see* database
- languages
 - multiple, 83–85
 - possible changes, 81
 - recommendations, 81
- latexmk, 100
- \mancite, 59
- multiple bibliographies
 - considered harmful, 76
 - document-based division, 72–74
 - refsection, 73
 - refsegment, 73
 - general types, 67
 - source-based division, 68–72
 - automation, 71
 - by type, 68
 - categories, 70
 - filters, 71
 - keywords, 69
 - printing, 68
 - selection, 68
- name, double-barrelled, 19
- names
 - database, in, *see* database
- Natbib, 14
- \newrefsection, 73
- \newrefsegment, 73
- \nocite, 59
- options, 34, 36
- pagination
 - in postnote, 57
- \parencite, 52
- pinpoints, *see* citation, postnote
- \pno, 56
- postnote, 51
- \ppno, 56
- prenote, 51
- \printbibliography, 61, 68
- \psq, 57
- \psqq, 57
- punctuation, 34, 91
 - after ‘In’, 92
 - author/year, 92
 - before postnote, 91
 - colon, 34
 - comma, 34
 - exclamation mark, 34
 - multiple citations, 92
 - period, 34
 - question mark, 34
 - quotation marks, 95
 - semicolon, 34
 - within entry, 94
- refsection
 - database, 73
 - general description, 72
- refsection environment, 73
- refsegment
 - database, 73
 - general description, 72
- refsegment environment, 73
- sets of sources cited together, *see* entryset
- shorthand, 24–25
 - list of shorthands, 64
- sorting
 - alphanumeric labels, 78
 - altering order, 79

- citation order, 78
- debug, 78
- general, 77
- in .bib file order, 78
- language specific, 85
- name/title/year, 78
- name/year/title, 78
- none, 78
- pre-sorting, 79
- predefined schemes, 78
- sortkey as last resort, 80
- texttsortname, 78
- texttsorttitle, 79
- special cases, 78
- year/name/title, 78
- sourcemap directive, 32, 36
- spaces, 34
- strings
 - commonly used, 34
 - styles
 - overview, 37–38
 - \supercite, 54
 - Texify, 101
 - \textcite, 53
 - title
 - citing title, 54
 - title, capitalization, 20
 - unicode
 - database, in, 17
 - URL
 - citing, 55
 - database, in, *see* database
 - year, *see* date