5 Textual tools

Every text-handling system needs to support a repertoire of tools for doing things with text. LaTeX implements many dozens, of which a small selection of the most frequently used is given here:

- footnotes and end-notes;
- marginal notes;
- cross-references, both normal ones and bibliographic citations;
- indexes and glossaries;
- typesetting in multiple columns.

5.1 Footnotes and end-notes

The command `\footnote{Like this}`,[1] followed by the text of the footnote in curly braces, will produce an auto-numbered footnote with a raised small number where you put the command,[Like this.] and the numbered text automatically printed at the foot of the page. The number is reset to 1 at the start of each

[1] Like this.

chapter (but there are packages to override that and make them run continuously throughout the document, or even restart at 1 on each page or section).

LaTeX automatically creates room for the footnote, and automatically reformats it if you change your document in such a way that the point of attachment and the footnote would move to the next (or preceding) page.

Footnotes in titling and sectioning commands (`\title`, `\caption`, and `\author`; `\chapter`, `\section`, etc) are regarded as Bad Style, and you SHOULD try to avoid them. If you can't, in `\title` and `\author` you MUST prefix the `\footnote` command with `\protect` to prevent it being taken as part of the text. In `\caption`, `\chapter`, `\section`, etc, you MUST use the optional argument to provide the un-footnoted version for the ToC (see section 2.6 on page 56).

Depending on the book or journal style, footnotes in titles may need to produce the symbols *, †, ‡, §, ¶, ||, **, ††, and ‡‡ instead of the values 1–9 (and an error message for the tenth such footnote). In accordance with standard publishing practice, footnotes inside a *minipage* environment (see section 4.6.1 on page 126) produce lettered notes instead of numbered ones, and they get printed at the bottom of the minipage, *not* the bottom of the physical page (but this too can be changed).

There is a package (endnote) to hold over your footnotes and make them print at the end of the chapter instead or at the end of the whole document, and there is a package (fnpara) to print many short footnotes in a single footnoted paragraph so they take up less space. It is also possible to have several separate series of footnotes active simultaneously, which is useful in critical editions or commentaries: for example, a numbered series for the original author's original footnotes; a lettered series for footnotes by subsequent commentators or authorities in later editions; and a roman-numeral series for your own footnotes. Note that some disciplines put their bibliographic references in footnotes (notably Historians) and they even call bibliographic references 'footnotes' (see section 5.3.2.3 on page 146).

It is also possible to format footnotes within footnotes, although this is considered Really Bad Style.

Verbatim inside footnotes

Because LaTeX reads the whole footnote before doing anything with it, you can't use the `\verb` (inline verbatim) command inside footnotes on its own: either use the `\VerbatimFootnotes` command from the fancyvrb package, or prefix `\footnote` with `\protect`, or use (abuse?) the `\url` command instead (which you should be using for Web and email addresses in any case).

If your footnotes are few and far between, you may want to use the sequence of footnote symbols above instead of numbers. You can do this by redefining the output of the footnote counter to be the `\fnsymbol` command (with the `footnote` counter as its argument) in the Preamble of your document:

```
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
```

There are also ways to refer to a footnote, and to defer the positioning of the footnote if it occurs in a float like a Table or Figure, where it might otherwise need to move to a different page, but these techniques are out of scope here.

5.2 Marginal notes

You can add marginal notes to your text instead of footnotes. or as well as You need to make sure that you have a wide-enough margin, of course: use the geometry package (see section 3.1.2 on page 67) to allocate enough space, otherwise the notes will be too cramped.

There are several packages to help with formatting marginal notes, but you can also define one yourself. Add this new command to your Preamble:

```
\newcommand{\marginal}[1]{%
  \leavevmode\marginpar{\tiny\raggedright#1\par}}
```

Then you can use `\marginal{Some text}` where you need it. Be careful, however, because marginal notes are aligned with the line where the command starts, so a very long one followed too closely by another will cause LaTeX to try and adjust the position so they don't overlap.

We're jumping ahead a bit here, as we haven't covered how to define your own commands yet. I won't even try to explain it here, although the astute reader can probably deduce it by inspection. See Chapter 7 starting on page 203 for more information about making up your own commands.

5.3 References and citations

This is one of the most powerful features of LaTeX. As we mentioned when discussing Figures and Tables, you can label any point in a document with a name you make up, so that you can refer to it by that name from anywhere else in the document (or even from another document) and LaTeX will always work out the right cross-reference number for you, no matter how much you edit the text or move it around.

As we well see later, a similar method is also used to cite documents for a bibliography or list of references, and there are packages to sort and format these in the correct style for different journals or publishers.[2]

5.3.1 Cross-references

You label the place in your document you want to refer *to* by adding the command `\label` followed by a short name you make up, in curly braces:[3] exactly as we did for labelling Figures and Tables in section 4.2.2 on page 98.

```
\section{New Research}
\label{newstuff}
```

[2] Be aware that in some disciplines where cross-references are not much used, the word 'references' may be used to mean 'bibliographic references'.

[3] This section is labelled `normalxref`, for example.

You can then refer to this place from anywhere in the same document with the command \ref followed by the name you used, eg

- Note the use of the unbreakable space (~) between the \ref and the word before it. This prints a space but prevents the line ever breaking at that point, should it fall close to the end of a line when being typeset.
- The \S command can be used if you want the section sign § instead of the word 'section' (there is also a \P command that produces the paragraph sign or pilcrow ¶).

```
In section~\ref{newstuff} there is a list of recent
projects.
```

In section 13.5 there is a list of recent projects.

Labels MUST be unique (that is, each value MUST occur only *once* as a label within a single document), but you can have as many references to them as you like. If you are familiar with HTML, this is the same concept as the internal linking mechanism using # labels (or IDs in XHTML or HTML5).

Labels in normal text: If the label is in normal text, as above, the reference will give the current chapter/section/subsection/appendix number (depending on the current document class).[4]

Labels in Tables or Figures: If the label is inside a Table or Figure, the reference provides the Table number or Figure number prefixed by the chapter number (remember that in Tables and Figures the \label command MUST come *after* the \caption command).

The \ref command does not produce the word 'Figure' or 'Table' for you: you have to type it yourself, or use the varioref package which automates it.

[4] So I can refer here to the label of this section as \ref {normalxref} and get the value 'Section 5.3.1 on the facing page'.

Labels in lists: A label in an item in an enumerated list will provide the item number. In other lists its value is null or undefined.

Labels elsewhere: If there is no apparent countable structure at the point in the document where you put the label (in a bulleted list, for example), the reference will be null or undefined.

The command `\pageref` followed by any of your label values will provide the page number where the label occurred, instead of the reference number, regardless of the document structure. This makes it possible to refer to something by page number as well as by its `\ref` number, which is useful in very long documents like this one (varioref automates this too).

Process twice!

LaTeX records the label values each time the document is processed, so the updated values will get used the *next* time the document is processed. You therefore need to process the document one extra time before final printing or viewing, if you have changed or added references, to make sure the values are correctly resolved. Most LaTeX editors handle this automatically by typesetting the document twice when needed.

Unresolved references are printed as two question marks, and they also cause a warning message at the end of the log file. There's never any harm in having `\label`s you don't refer to, but using `\ref` when you don't have a matching `\label` is an error, as is defining two labels with the same value.

5.3.2 Bibliographic references

The mechanism used for references to reading lists and bibliographies is very similar to that used for normal cross-references. Instead of using `\ref` you use `\cite` or one of the variants explained in below; and instead of `\label`, you attach a label

value to each of the reference entries for the books, articles, reports, etc that you want to cite. You keep these reference entries in a ***bibliographic reference database*** that uses the BibTeX data format (see below).

Bibliographic reference databases

Although it is possible to type the details of each reference manually, it's much easier to use a program designed for the purpose. There are several available (see Wikipedia's list), including Zotero and Mendeley. Both are open-source, but *Mendeley* was recently bought out by Elsevier; although it was still free of charge at the time of writing there have been concerns about Elsevier's use of your data. Their features vary but *Zotero*'s primary benefit is that it can grab bibliographic metadata from web pages, so that you don't have to type it in. Both can extract the metadata from the PDFs of articles you download from journal sites. And both can export the data in BibTeX format, which is essential.

Once your data is in BibTeX format, you can manage your collection of references with any of the many free BibTeX-based database programs such as JabRef (see Figure 5.1 on page 149).

Endnote and Reference Manager are commercial products which do not use the BibTeX data format, but which can export the Research Information Systems (RIS) format which *Zotero*, *JabRef*, and *Mendeley* can all import.

You add your entries to whichever system you choose, usually by downloading references from an online database like *Web of Science*, *JSTOR*, *PubMed* etc, or by using *Zotero* or similar to gather the entry from a web page (you can also type references in by hand). *JabRef* lets you click an icon or menu entry and the LaTeX citation command will be inserted into your document editor at the cursor location.

This does away with the time needed to maintain and format references each time you cite them, and dramatically improves accuracy. It means you only ever have to enter the bibliographic

details of your references once, and you can then cite them in any document you write, and the ones you cite will get formatted automatically to the style you specify for the document (eg Harvard, Oxford, Institute of Electrical and Electronics Engineers (IEEE), Vancouver, Modern Language Association (MLA), American Psychological Association (APA), etc).

5.3.2.1 Choosing between BibTeX and *biblatex*

LaTeX has two systems for doing citations and references, BibTeX (old) and biblatex (new): both of them use the same file format, also called BibTeX. Both support the four common ways of indicating a citation: author-year, numeric, abbreviated alphabetic, and footnoted; plus a wide range of others.

BibTeX: the older BibTeX has been in use for many decades and is still specified in some publishers' document classes, especially for journal articles and books, and particularly those which have not been updated for a long time. While it will continue to work, it has several drawbacks:

1. it doesn't handle non-ASCII characters easily, so accents and non-Latin words are a problem;
2. the same applies to the sort-and-extract program it uses (also called *bibtex*);
3. the style format files (`.bst` files) are written in BibTeX's own rather strange, unique, and largely undocumented language, making it extremely difficult to modify them or write new ones;
4. many of the style format files are now very old and out of date;
5. the range of data fields in references is limited and also out of date.

biblatex the newer biblatex system is now a well-established LaTeX package to replace almost all of old BibTeX. The main advantages are:

1. it uses the same `.bib` files as for old BibTeX, but lets you use many new document types and data field names.

2. it works with UTF-8, so non-ASCII, non-Latin, and other writing systems are handled natively when using XeLaTeX or LuaLaTeX.
3. there is a new sort-and-extract program (*biber*) to replace *bibtex*, which also handles UTF-8 natively.
4. the style format files are written entirely in LaTeX syntax, and are under active development, so updating or writing layout formats it is much easier than with BibTeX.
5. it supports the four popular citation formats listed above natively, without the need for additional packages.

The only current drawback is that there are still a few uncommon and less-used reference formats that are still only supported in BibTeX and not yet available in biblatex. If you are required to use one of these, you are going to be stuck with BibTeX (unless you'd like to write a new biblatex add-on to handle it).

The biblatex package with the *biber* program is therefore recommended, especially with XeLaTeX or LuaLaTeX. From here on, I shall be concentrating on biblatex.

5.3.2.2 The BibTeX file format

The same file format for BibTeX files is used for both BibTeX and biblatex, regardless of whether you use *biber* or *bibtex*, so if you have existing BibTeX files, they will continue to work, but it's a good idea to update old files with some of the more accurate field names provided by biblatex.

The file format is specified in the original BibTeX documentation (look on your system for the file `btxdoc.pdf`). The biblatex package and its updated style formats provide many more fields and document types than we can describe here.

Each BibTeX entry starts with an `@` sign and the type of document (eg `article`, `book`, etc), followed by the whole entry in a single set of curly braces. The first value MUST be a unique BibTeX key (label) that you make up, which you will use to cite the reference with; followed by a comma:

```
@book{fg, ... }
```

Then comes each field (in any order), using the format:

```
fieldname = {value},
```

There MUST be a comma after each line of an entry *except the last line*:

```
@book{fg,
  title     = {{An 'Innkeepers Diary}},
  author    = {John Fothergill},
  edition   = 3,
  publisher = {Penguin},
  year      = 1929,
  address   = {London}
}
```

Some TEX-sensitive editors have a BIBTEX mode which understands these entries and provides menus, templates, and syntax colouring for writing them. The rules are:

- ☐ There MUST be a comma after each line of an entry *except the last line*;
- ☐ There MUST NOT be a comma after the last field in the entry (*only* — eg after {London} in the example);
- ☐ Some styles recapitalise the title when they format: to prevent this, enclose the title in double curly braces as in the example;
- ☐ You MUST use extra curly braces to enclose multi-word surnames, otherwise only the last will be used in the sort, and the others will be assumed to be forenames, for example the British explorer can be sorted under T as

  ```
  author = {Ranulph {Twisleton Wykeham Fiennes}},
  ```

- ☐ Multiple authors MUST go in a single author field, separated by the literal word and (see example below);

- Values which are purely numeric (eg years) may omit the curly braces;
- Months and editions MUST be numbers (and may therefore omit the curly braces); DO NOT include ordinal indicators like `th` or `st`;
- Fields MAY occur in any order but the format MUST otherwise be strictly observed;
- Fields which are not used do not have to be included (so if your editor automatically inserts them as blank or prefixed by `OPT` [optional], you MAY safely delete them as unused lines).

There is a required minimum set of fields for each of a dozen or so types of document: book, article (in a journal), article (in a collection), chapter (in a book), thesis, report, conference paper (in a Proceedings), etc, exactly as with all other reference management systems. These are all (entry types and entry fields) listed in detail in the biblatex documentation (Lehman et al. 2015, sections. 2.1 & 2.2, 8).

Here's another example, this time for a book on how to write mathematics — note the multiple authors separated by `and`. Long entries can spread over several lines: the extra spaces and line-breaks are ignored, so long as the value ends with the matching curly brace (and comma, if needed).[5]

```
@book{mathwrite,
  author    = {Donald E Knuth and Tracey
               Larrabee and Paul M Roberts},
  title     = {{Mathematical Writing}},
  publisher = {Mathematical Association of America},
  address   = {Washington, DC},
  series    =  {MAA Notes 14},
  isbn      = {0-88385-063-X},
  year      = 1989
}
```

[5] The major differences between BibTeX's use of these files and biblatex's use of them is that biblatex allows many more different types of fields, and is generally more up-to-date; and *biber* sorts UTF-8 correctly, and is more configurable.

Every reference in your reference database MUST have a unique key value (label or ID): you can make this up, just like you do with normal cross-references, but some bibliographic software automatically assigns a value, usually based on an abbreviation of the author and year. These keys are for *your* convenience in referencing: in normal circumstances your readers will not see them. You can see these labels in the right-hand-most column and at the bottom of the screenshot in Figure 5.1 on page 149, and in the examples above. You use these labels in your documents when you cite your references (see section 5.3.2.3).

There are many built-in options to the biblatex package for adjusting the citation and reference formats, only a few of which are covered here. Read the package documentation for details: it is possible to construct your own style simply by adjusting the settings, with no programming required (unlike the older BibTeX styles, which are written in a programming language used nowhere else).

Many users keep their BibTeX files in the same directory as their document[s], but it is also possible to tell LaTeX and BibTeX that they are in a different directory. This is a directory specified by the `$BIBINPUTS` shell or environment variable set up at installation time. On Unix & GNU/Linux systems (including Apple Macintosh OS X), and in TeX Live for Windows, this is your TeX installation's `texmf/bibtex/bib` directory — the same one that old-style BibTeX `.bst` style files are kept in — but to avoid permission conflicts you should use your Personal TeX Directory and create a subdirectory of the same name in there for your own `.bib` files. MiKTeX also uses the same `$BIBINPUTS` variable, but it is not set on installation: you need to set it using the Windows Systems Settings (see for example `www.computerhope.com/issues/ch000549.htm`).

5.3.2.3 Citation commands

The basic command is `\cite`, followed by the label of the entry in curly braces. You can cite several entries in one command: separate the labels with commas.

```
\cite{fg}
```

```
\cite{bull,davy,heller}
```

For documents with many citations, use the Cite button or menu item in your bibliographic reference manager, which will insert the relevant command for you (you can see it activated for the *TEXStudio* editor in below).

How the citation appears is governed by two things:

1. the reference format (style) you specify in the options to the biblatex package (see section 5.3.2.4 on page 150);
2. the type of citation command you use: \cite, \textcite, \parencite, \autocite, \footcite, etc, as shown below.

There are four built-in formats in biblatex:

authoryear: There are two basic types of author–year citation:

author as text, year in parentheses: used in phrases or sentences where the name of the author is part of the sentence, and the year is only there to identify what is being cited; this command is \textcite{fg}

. . . as has clearly been shown by Fothergill (1929).

This is sometimes called 'author-as-noun' citation.

whole citation in parentheses: used where the phrase or sentence is already complete, and the citation is being added in support: this command is \parencite{fg}

. . . as others have already clearly shown (Fothergill 1929).

The references at the end of the document are sorted into surname order of the first author, and by year after that.

numeric: This format is popular in some scientific disciplines where \cite produces just a number in square brackets, eg [42]. The references at the end of the document are usually numbered *either* in order of citation *or* in order of first surname;

alphabetic: This format is also popular in some scientific disciplines and \cite produces a three- or four-letter

abbreviation of the author's name and two digits of the year, all in square brackets, eg [Fot29]. The references at the end of the document are sorted using abbreviated key value as their label. This format is also called 'abbreviated'.

footnoted : Footnoted citations are common in History and some other Humanities disciplines, to the extent that scholars in these fields actually call their references 'footnotes'.[6] The command `\footcite` produces a superscript number like an ordinary footnote, and a short reference at the foot of the page. It is only relevant when using author-year styles (in numeric style it would just produce the reference number at the foot of the page, which would be misleading). The references at the end of the document are given in full, and are usually sorted alphabetically by first surname.

To direct your reader to a specific page or chapter, you can add a prefix and/or a suffix as optional arguments in square brackets before the label.

```
...as shown by \textcite[p 12]{mathwrite}.
```

A prefix gets printed at the start of the citation and the suffix gets printed at the end, but all still within the parentheses, if any. As they are both optional arguments, and as suffixes are far more common than prefixes, when only one optional argument is given, it is assumed to be the suffix. The example above therefore produces:

...as shown by Knuth, Larrabee and Roberts (1989, p. 12).

There are many variant forms of the citation commands, either for specific styles like Chicago, Vancouver, Harvard, IEEE, APA, MLA, etc; or for grammatical modifications like capitalising name prefixes, omitting the comma between name and year, or adding multiple notes; or for extracting specific fields from an entry (eg `\titlecite`). If you have requirements not met by the formats described here, you can find them in the documentation for the biblatex package.

[6] This can be confusing to outsiders: it's not clear how they refer to conventional footnotes, or if they even use them.

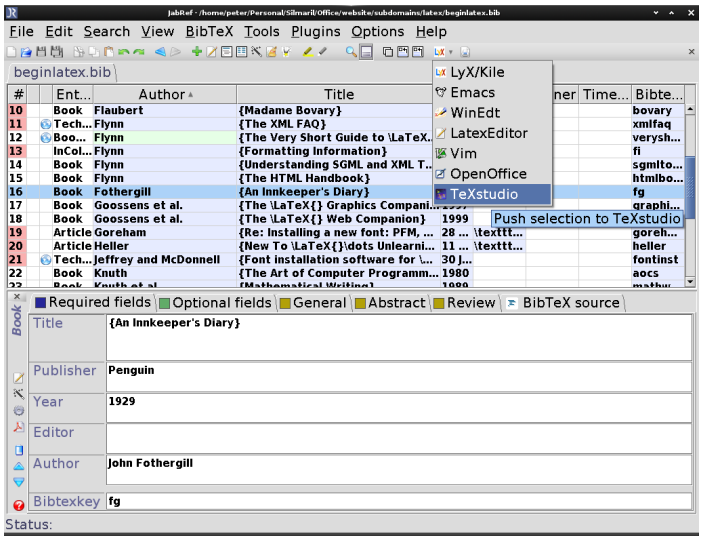
Table 5.1 – Built-in biblatex style commands and formats

Style	Command	Result
authoryear	\parencite{fg}	(Fothergill 1929)
authoryear	\textcite{fg}	Fothergill (1929)
authoryear	\footcite{fg}	[1]
numeric	\cite{fg}	[42]
alphabetic	\cite{fg}	[Fot29]
authoryear	\cite{fg}	Fothergill 1929

[1] Fothergill 1929.

Modern Language Association (MLA) citation is a special case, as it omits the year and instead REQUIRES the location of the citation within the document (eg the chapter, section, page, or line). It may include the title, if there would otherwise be ambiguity. The biblatex format for MLA citation handles the context-dependent formatting with the command \autocite.

Figure 5.1 – JabRef displaying a file of references, ready to insert a citation of Fothergill's book into a LaTeX document being edited with TeXStudio



Your reference management software will have a display something like above (details vary between systems, but they all do roughly the same job in roughly the same way), showing all your references with the data in the usual fields (title, author, date, etc).

Your database, which contains all your bibliographic data, MUST be saved or exported as a BibTeX format (`.bib`) file from your reference management software (*JabRef* uses this format automatically), It looks like the examples in section 5.3.2.2 on page 143. Your `.bib` file works with both biblatex *and* BibTeX, but biblatex provides more field types and document types so that your references can be formatted more accurately.

If your bibliographic management software doesn't save BibTeX format direct, save your data in RIS format, then import the `.ris` file into *JabRef* and save it as a `.bib` file from there.

Cheatsheet

Clea F Rees has written an excellent cheatsheet with virtually everything on it that you need for quick reference to using biblatex. This is downloadable as the package biblatex-cheatsheet from CTAN.

5.3.2.4 Setting up biblatex with *biber*

For more complex citation requirements, you may need to set up your document with the following packages:

1. the babel or polyglossia package with appropriate languages, *even if you are only using one language*. The default language is American English, so there are commands to map this to other language variants (the example below shows this for British English);

2. the csquotes package, which automates the use of quotation marks around titles or not, depending on the type of reference;

Exercise 25 – Using biblatex

1. Use your bibliographic database program (eg *JabRef* or similar) to create a file with your references in it (see section 5.3.2.2 on page 143). Make sure each entry has a unique short keyname (**bibtexkey**) to make citations with;
2. Save the file with a name ending in `.bib` (eg `myrefs.bib`) in the same folder as your LaTeX document;
3. Add these two lines to the Preamble of your LaTeX document:

```
\usepackage[backend=biber,style=authoryear]{biblatex}
\addbibresource{myrefs.bib}
```

4. In the body of your document, where you want to cite a work, use `\parencite{keyname}` or `\textcite{keyname}` as appropriate;
5. Just before the end of your document, add the command `\printbibliography` at the point where you want the bibliography printed ;
6. Typeset the document: run *xelatex*, then *biber*, and then *xelatex* again (most editors automate this);
7. When you've got the citations and references working, read the BibTeX documentation.

3. the biblatex package itself, specifying the *biber* program and the style of references you want, either `numeric`, `alphabetic`, or `authoryear`; or a publisher's style; and any options for handling links like DOIs, URIs, and ISBNs;

4. the language mapping command, if needed (see the documentation for the style you have chosen to find out if you need this);

5. finally, the name of your BibTeX file[s] (see the panel 'Bibliographic reference databases' on p. 141bibdb) with one or more \addbibresource commands.

```
\usepackage[frenchb,german,british]{babel}
\usepackage{csquotes}
\usepackage[backend=biber,doi=true,isbn=true,
            url=true,style=apa]{biblatex}
\DeclareLanguageMapping{british}{british-apa}
\addbibresource{myrefs.bib}
```

At the end of your document you can then add the command \printbibliography (or elsewhere that you want the full list of references you have cited to be output). See section 5.3.2.5 for details of how LaTeX produces the references.

Versions of *biber* and biblatex

One critically important point to note is that biblatex and *biber* are step-versioned; that is, each version of the biblatex package only works with a specific version of the *biber* program. There is a table of these dependencies in the biblatex documentation PDF. If you manually update biblatex for some reason (perhaps to make use of a new feature), you MUST also update your copy of *biber* to the correct version, and *vice versa*, otherwise you will not be able to produce a bibliography.

5.3.2.5 Producing the references

Because of the record→extract→format process (the same as used for cross-references), you will get a warning message about 'unresolved references' the first time you process your document after adding a new citation for a previously uncited work. LaTeX inserts the label of the reference in bold as a marker or placeholder until you run *biber* and re-typeset the document. This is why most editors have a Build function to do the job for you.

This function should therefore handle the business of running *biber* and re-running XeLaTeXfor you. If not, here's how to do it manually in a Command window: you run XeLaTeXthen run *biber* to extract and sort the details from the BibTeX file, and then run XeLaTeXagain:

```
xelatex myreport
biber myreport
xelatex myreport
```

In practice, authors tend to retypeset their documents from time to time during writing anyway, so they can keep an eye on the typographic progress of the document. So long as you remember to click the Build or equivalent button after adding a new \cite command, all subsequent runs ofXeLaTeXwill incrementally incorporate all references without you having to worry about it.

If you work from the command line, the *latexmk* script automates this, running *bibtex* or *biber* and re-running LaTeX again when needed.

5.4 Indexes and glossaries

Indexes and glossaries are tools for directing or helping the reader. Any book or report sized document should have an index, although they are uncommon in theses. Glossaries are usually only needed where there is a substantial number of technical terms needing formal definition and cross-referencing.

5.4.1 Indexes

LaTeX has an automated indexing facility which uses the standard *makeindex* program for sorting and collation. To use indexing, use the package makeidx and include the \makeindex command in your Preamble to initialise the index:

```
\usepackage{makeidx}
\makeindex
```

When you want to index something, use the command \index followed by the entry in curly braces, as you want it to appear in the index, in one of the following formats:

Plain entry: Typing \index{beer} will create an entry for 'beer' with the current page number;

Subindex entry: For an entry with a subentry use an exclamation mark to separate them: \index{beer!lite}. You can create another level as well, so you can have subsubentries like \index{beer!lite!American};

Cross-references: 'See' entries are done with the vertical bar (one of the rare times it does *not* get interpreted as a math character): \index{Microbrew|see{beer}};

Font changes: To change the typographic style of an entry, use the @-sign followed by a font change command:

```
\index{beer!Rogue!Chocolate Stout@\textit{Chocolate Stout}}
```

This example indexes *Chocolate Stout* as a third-level entry and italicises it at the same time. Any of the standard \text... font-change commands work here: see Table 6.3 on page 190 for details.

You can also change the font of the page number on its own, for example for first-usage references, by using the vertical bar in a similar way to the 'see' entries above, but instead of 'see', substituting a font-change command name alone (*without* backslash or curly braces) such as textbf for a page number in bold (see the index):

```
\index{beer!Rogue!Chocolate Stout|textbf}
```

Out of sequence: The same method can be used as for font changes, but using the alternate index word instead of the font command name, so \index{Oregon Brewing Company@Rogue} will add an entry for 'Rogue' in the 'O' section of the index, as if it was spelled 'Oregon Brewing Company'.

When the document has been processed through LaTeX it will have created a `.idx` file, which you run through the *makeindex* program by clicking the Index button or menu entry in your editor, or by typing the `makeindex` command followed by your document name without the `.tex` filetype. Most editors will do this automatically as they process your document if they spot that you have generated a `.idx` file.

The *makeindex* program creates a file with the same name as your document, but with the `.ind` filetype, containing the sorted index. This is what gets used by the command `\printindex` which you put at the end of your document, where you want the index printed. The default index format is two columns with a space between letters of the alphabet. The Unix manual page[7] for the *makeindex* program has details of how to add letter headings to each alphabet group.

5.4.2 Glossaries

Glossaries can be done in a similar manner to indexes, using the command `\makeglossary` in the Preamble and the command `\glossary` in the same way as `\index`. There are some subtle differences in the way glossaries are handled: both the books by Lamport (1994) and by Mittelbach et al. (2004) duck the issue, but there is some documentation on `glotex` on CTAN. There is also a gloss package based on BibTeX which uses `\gloss` in the same way as `\cite`.

However, by far the best way is to use the glossaries package (not glossary, which is obsolete; and not gloss either). This is a relatively complex package, as glossaries are a relatively complex tool, but there is extensive help in the documentation. It requires the *makeglossaries* script from CTAN (there is also a *makeglossariesgui* Java GUI).

Basically, you need to create a set of definitions, one per item to be glossed, using the `\newglossaryentry` command. Think

[7] On Unix & GNU/Linux systems (including Apple Macintosh OS X, just type the command `man makeindex`; the page is also available in many reference sites on the web.

of this as being the equivalent to a reference entry in your bibliography.

```
\newglossaryentry{esis}{name={ESIS},description={The
  \textbf{Element Structure Information Set} of a
  marked-up document, originally defined for SGML
  (replaced for XML using W3C Schemas by the
  Post-Schema-Validation InfoSet, PSVI). See
  \url{http://xml.coverpages.org/WG8-n931a.html}}}
```

This specifies *a*) the label you will use (`esis`); *b*) the name of the item as it will be printed (`ESIS`); and *c*) the textual description to go in the glossary. Probably the bext place to put these is in a separate file like `mygloss.tex`, which you can get LaTeX to read with an `\input{mygloss.tex}` command in your Preamble.

You can then use the `\gls` command in your text to produce the printable name of any entry, using the label to refer to it, so `\gls{esis}` will produce 'ESIS'. It is possible to use or define variant commands to handle references at the start of a sentence (where you need a capital letter if the name is not an acronym), and grammatical alteration like unusual plurals or forms ending in '—ing'.

At the end of your document, where you want the glossary printed, you use the command `\printglossaries`. The glossary needs to be processed separately from the main document, using the *makeglossaries* script, exactly the same way as you do for *biber*, *bibtex*, and *makeindex*. The Build function of your editor should do this for you, or you can use a Makefile or a utility like *latexmk*. As with all these tools, there are many more facilities built into them: read the documentation.

The acronym package can also be used to create a kind of glossary containing a list of acronyms and their expansions, as well as driving the use of expansion on first mention. However, the glossaries package now contains built-in support for acronyms.

For Linnaean taxa, use the biocon package, which automates the way in which plant and animal species names and ranks are

typeset and referred to in formal writing. There are other more complex packages for managing taxonomies.

5.5 Multiple columns

Use the multicol package: the environment is called *multicols* (note the plural form) and it takes the number of columns as a second argument in curly braces:

```
\usepackage{multicol}
...
\begin{multicols}{3}
...
\end{multicols}
```

LaTeX has built-in support for two-column typesetting via the *twocolumn* option in the standard Document Class Declarations, but it is relatively inflexible in that you cannot change from full-width to double-column and back again on the same page, and the final page does not balance the column heights. However, it does feature special *figure** and *table** environments which typeset full-width figures and tables across a double-column setting.

The more extensive solution is the multicol package, which will set up to 10 columns, and allows the number of columns to be changed or reset to one in mid-page, so that full-width graphics can still be used. It also balances the height of the final page so that all columns are the same height — if possible: it's not always achievable — and you can control the width of the gutter by setting the \columnsep length to a new dimension. There is a *multicols** environment which does not try to balance the columns.

Multi-column work needs some skill in typographic layout, though: the narrowness of the columns makes typesetting less likely to fit smoothly because it's hard to hyphenate and justify well when there is little space to manœuvre in.