`biblatex` variations

Ulrike Fischer

Abstract

I show three small examples of using the `biblatex` package for more than printing bibliographies: we will redefine bibliography drivers, define new cite commands and declare new entry types to create qrcodes, insert PDF files and manage addresses.

Remark

In April I gave a talk at the DANTE e.V. meeting about `biblatex` variations and later wrote an article for the proceedings in *Die TEXnische Komödie*. This is more or less a translation of that article. I didn't adapt the examples, so they are still in German.

ad hoc small-scale databases

I have always been interested in the handling of small databases. I wrote my first articles in *DTK* ([1, 2]) about a mail merging system that I developed to handle around 50 addresses. And later on I regularly had to find ways to automatically process small numbers of data records without too much fuss.

For such small scale databases, the `bib` is an interesting option — at least on the input side. Looking at it without the prejudice "that is something for bibliographies only" — Listing 1 shows such a typical `bib` file, that I will use in this article — one can see that it has several features making it suitable for *ad hoc* small scale databases:

- One can mix different datatypes in one file.
- One can easily create new datatypes.
- One can easily add new fields.
- Fields without values can be (should be) omitted. This saves space.
- The records can be created, changed and read with any editor, but there are also good GUIs (e.g. JabRef), so the database can be edited by people who don't know LaTeX.
- With `@string` one can define variables.
- It is possible to define relations between records, e.g. with `crossref`.

So, it is easy to create a small database but ... how should one process and output the records? Before `biblatex` this was not usually feasible. Creating a suitable `bst` file was a difficult and time-consuming task. But *with* `biblatex` this has completely changed. Now it is possible, with very little effort, to output the records in various ways. The following examples are meant to demonstrate this. They will show various methods one can use. The

Listing 1: The example bib file `vortrag.bib`

```
@termin{dante2014,
 title={Biblatex-Variationen},
 date ={2014-04-11},
 time ={15.15},
 location={Heidelberg}}

@online{dante,
 title={Internetseite dante e.V.},
 url  ={http://www.dante.de}}

@online{heidelberg,
 title={Stadt Heidelberg},
 url  ={http://www.heidelberg.de}}

@adresse{max,
 name    ={Muster, Max},
 strasse ={Im Versuchsweg 10},
 ort     ={Testgelände},
 plz     ={X01234},
 gender  ={sm}}

@adresse{eva,
 name    ={Muster, Eva},
 strasse ={Im Versuchsweg 10},
 ort     ={Testgelände},
 plz     ={X01234},
 gender  ={sf}}

@article{input1,
 author={Fischer, Ulrike},
 title ={Erster Text},
 journal={Beispiele},
 date  ={2012-04-08},
 url   ={inputtext1.pdf}}

@article{input2,
 author={Fischer, Ulrike},
 title ={Zweiter Text},
 journal={Beispiele},
 date  ={2013-02-07},
 url   ={inputtext2.pdf}}

@book{gambol,
author={Gambolputty de von Ausfern-
  ↪schplenden-schlitter-crasscrenbon,
  ↪Johann},
title={Titel},
year={1970}}

@book{dante2007,
author = {Dante Alighieri},
title = {Die Göttliche Kommödie},
gender = {sm},
location = {Stuttgart},
year = {2007},
translator = {Hermann Gmelin}}
```

Listing 2: The creation of QR-codes

```
1  % Compile with XeLaTeX
2  \documentclass{article}
3  \usepackage[margin=0.05in,textwidth=1.9in,
   ↪textheight=1.9in,paperwidth=2in,
   ↪paperheight=2in]{geometry}
4  \usepackage{xcolor}
5  \usepackage{pst-barcode}
6  \usepackage{fontspec}
7  \usepackage{biblatex}
8  \addbibresource{vortrag.bib}
9
10 \defbibenvironment{qrcode}{\centering}{}{}
11
12 \DeclareBibliographyDriver{online}{%
13   \begin{minipage}[c][1.9in]{1.9in}
14     \centering
15     \printtext{\thefield{entrykey}}\\[2ex]
16     \printfield{title}\\[2ex]
17     \printfield{url}
18  \end{minipage}
19  \newpage
20  \begin{pspicture}(1.9in,1.9in)
21     \label{\thefield{entrykey}}%
22     \psbarcode[linecolor=red]{\thefield{url
   ↪}}{width=1.9 height=1.9}{qrcode}%
23    \end{pspicture}%
24  \newpage}
25
26 \begin{document}
27 \nocite{*}
28 \printbibliography[env=qrcode,type=online,
   ↪heading=none]
29 \end{document}
```

examples are kept as simple as possible. In larger databases one would likely need to add some security precautions, such as tests for empty fields.

1 Example 1: QR-codes

In this example, we first create a PDF file which contains the QR-code of the `url` field of every `@online` entry in a `bib` file. The QR-codes can then be inserted with `\includegraphics` in another document. As methodology, the redefinition of a *bibliography driver* is shown.

1.1 The creation of the QR-codes

Herbert Voß has shown how QR-codes can be created generally in a *DTK* article [3]. That method uses the package `pst-barcode`, meaning that one needs a TEX compiler which can handle PostScript; I usually use `xelatex`.

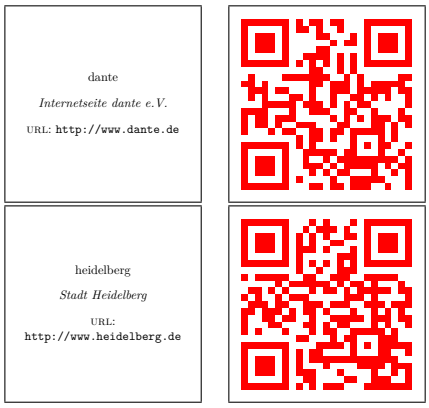

Figure 1: The PDF output pages with QR-codes

Listing 2 shows how one can create QR-codes from the `url` field of a `bib` file. The actual document body is very short (lines 26–29): all entries are cited with `\nocite{*}` and then the bibliography is printed with `\printbibliography`, which is given three options: `heading=none` suppresses the heading of the bibliography; `type=online` only outputs entries of type `@online`; and `env=qrcode` ensures that the bibliography is not a rather complicated list but the simple environment defined in line 10.

The start of the listing (lines 3–8) is basic: the page size is set to 1.9 in × 1.9 in, needed packages are loaded and the name of the `bib` file is declared.

Line 10 defines a simple `qrcode` environment, as the standard list environment would only insert unwanted spaces.

The core of the approach is in lines 12–24. They define how a `@online` entry is formatted in the bibliography. The code creates two pages for each entry.

The first page shows some information about the following QR-codes. This page is not required; I produce it only because it looks nice. The code use the `biblatex` commands `\printtext`, `\printfield` and `\thefield` to output fields from the bib record.

Lines 20–24 create the second page with the QR-code. A useful addition is the label, specified in line 21: an external document is then able to find the page with the QR-code of a specific `bib` key (the "entrykey"). In line 22 the QR-code is created. The url is inserted with `\thefield{url}`.

Compiling with `xelatex`–`biber`–`xelatex` results in a PDF file with four pages, shown in figure 1.

1.2 Using the QR-codes

Listing 3 shows how one can insert the QR-codes in other documents. The code uses the package `refcount` to convert a label to a number which can be used with the option `page` of `\includegraphics`.

Listing 3: Inserting the QR-codes

```
\documentclass[parskip=half-]{scrartcl}
\usepackage{graphicx,refcount,xr}
\externaldocument[qrcode-]
  {qrcodes-bib-erzeugen}
\begin{document}
\section*{QR-Codes laden}
\includegraphics[page=
    \getpagerefnumber{qrcode-dante}]
  {qrcodes-bib-erzeugen-dtk}
\quad
\includegraphics[page=
    \getpagerefnumber{qrcode-heidelberg}]
  {qrcodes-bib-erzeugen-dtk}
\end{document}
```

It also uses the package `xr` to access the labels of the document with the QR-codes. The code must be able to find the PDF and `aux` files of the external document with the QR-codes.

2 Example 2: Inserting PDF attachments

The second example inserts a PDF file in a document and writes information about this file to the table of contents. The method used is the definition of a new cite command. The example also shows that things do not always work as smoothly as one would wish.

Listing 4 shows the core idea: In lines 12–27 a new cite command with the name `\citeanlageX` is defined. Defining a cite command is a bit more complicated then defining a driver, as a cite command can have *lists* of entries as an argument.

In the so-called *loopcode* argument (lines 14–26) a new page is started and the counter for the attachment is advanced (line 14). Then in lines 15–22 an entry for the `toc` file is written. The content of this entry is the word "Anlage" followed by the number and a `\fullcite`.

In lines 23–26 the file from the `url` field is included with `\includepdf`. The existence of the file is checked first with `\IfFileExists`.

`\citeanlageX` does everything that is needed, but it has a flaw: it can lead to unwanted empty pages. The problem is that every `biblatex` cite command internally executes `\leavevmode`, and so can start a page. Together with the `\clearpage` there is then a page too many.

So I had to dig around a bit in the code to find the source of the `\leavevmode`. Happily it is easy to deactivate it locally. This is done in lines 29–36.

Listing 4: Inserting PDF attachments

```
1  \documentclass[parskip=half-,toc=flat]{
   ↪scrartcl}
2  \usepackage[utf8]{inputenc}
3  \usepackage[T1]{fontenc}
4  \usepackage[ngerman]{babel}
5  \usepackage[autostyle]{csquotes}
6  \usepackage{pdfpages}
7  \usepackage[style=authoryear]{biblatex}
8  \addbibresource{vortrag.bib}
9
10 \newcounter{anlage}
11
12 \DeclareCiteCommand{\citeanlageX}
13  {}
14  {\clearpage\refstepcounter{anlage}%
15   \addtocontents{toc}
16     {\protect\contentsline
17         {section}
18         {\protect\numberline{Anlage~\
   ↪theanlage}%
19          \protect\fullcite{\thefield{
   ↪entrykey}}%
20         }%
21       {\thepage}
22       {}}%
23   \IfFileExists{\thefield{url}}
24    {\includepdf[pages=-,fitpaper]{\thefield{
   ↪url}}}
25    {\par\textbf{Datei zu \thefield{entrykey
   ↪} wurde nicht gefunden!}%
26     \clearpage}}
27  {}{}
28
29 \makeatletter
30 \newcommand\citeanlage[1]{%
31  \begingroup
32    \let\blx@leavevmode\relax
33    \let\blx@leavevmode@cite\relax
34    \citeanlageX{#1}%
35  \endgroup}
36 \makeatother
37
38 \begin{document}
39 \tableofcontents
40
41 \citeanlage{input1} \citeanlage{input2}
42 \end{document}
```

Inhaltsverzeichnis

Anlage 1 Ulrike Fischer (2012). „Erster Text". In: *Beispiele*. URL: inputtext1.pdf 2

Anlage 2 Ulrike Fischer (2013). „Zweiter Text". In: *Beispiele*. URL: inputtext2.pdf 4

Figure 2: The table of contents created by Listing 4

Listing 5: The datamodel file `ufischer.dbx`

```
1  \DeclareDatamodelEntrytypes{adresse}
2
3  \DeclareDatamodelFields[type=list,
   ↪datatype=name]
4   {name}
5
6  \DeclareDatamodelFields[type=field,
   ↪datatype=literal]
7   {strasse,ort,plz}
8
9  \DeclareDatamodelEntryfields[adresse]{%
10   name,strasse,ort,plz,gender}
```

3 Example 3: Managing addresses

The third example is a bit more complicated. It shows how to manage addresses in a `bib` file. The new method shown this time is how the *datamodel* can be extended.

New entrytypes and fields should be declared in a `dbx` file, as shown in Listing 5. In line 1 a new entry type `adresse` is added and in lines 3–7 new fields for this type. One should also declare which fields can be used by an entry type. A new entry type can use known fields, such as the field `gender` on the last line.

Listing 6 shows how to use the new entry type. First, the new datamodel is loaded in line 7 with `datamodel=ufischer`.

Lines 10–14 declare a new bibliography driver `adresse` for the distribution list; it creates a simple, comma-separated list of the data.

Line 16 declares a *name format* for the greeting in the letter. It prints only the last name from a name.

In lines 18–22 a cite command for the greeting is defined: `\citeanrede`. It uses the gender field to decide if "Frau" or "Herr" should be printed before the name.

In lines 24–30 another cite command is defined: `\citeadresse`. This command is meant for the address window and prints the data line-by-line.[1]

Lines 32–43 show how the various commands can be used. The output can be seen in figure 3.

4 Finally: How to control the content of the bibliography?

When one starts to "misuse" bib entries for things other than standard citations, one quickly finds the need to prevent such entries from finding their way

[1] Many of the comment signs in the code are not needed but they do no harm, either.

Listing 6: How to use the type `@adresse`

```
1  \documentclass[parskip=half-,toc=flat,
   ↪fontsize=9pt,DIV  = 9,
2                paper=a5,pagesize,headings=
   ↪normal]{scrartcl}
3  \usepackage[utf8]{inputenc}
4  \usepackage[T1]{fontenc}
5  \usepackage[ngerman]{babel}
6  \usepackage[autostyle]{csquotes}
7  \usepackage[datamodel=ufischer,defernumbers
   ↪]{biblatex}
8  \addbibresource{vortrag.bib}
9
10 \DeclareBibliographyDriver{adresse}{%
11   \printnames{name}\setunit{\addcomma\
   ↪addspace}%
12   \printfield{strasse}\setunit{\addcomma\
   ↪addspace}%
13   \printfield{plz}\setunit{\addspace}\
   ↪printfield{ort}%
14   \usebibmacro{finentry}}
15
16 \DeclareNameFormat[adresse]{anrede}{#1}
17
18 \DeclareCiteCommand{\citeanrede}{}{%
19   \iffieldequalstr{gender}{sm}
20    {\printtext{Herr}}{\printtext{Frau}}%
21   \setunit{\addspace}\printnames[anrede]{
   ↪name}}
22   {}{}
23
24 \DeclareCiteCommand{\citeadresse}{}{%
25   \printtext{\par\noindent}%
26   \iffieldequalstr{gender}{sm}{\printtext{
   ↪Herrn}}{\printtext{Frau}}%
27   \setunit{\\}\printnames{name}%
28   \setunit{\\}\printfield{strasse}%
29   \setunit{\\}\printfield{plz}\setunit{\
   ↪addspace}\printfield{ort}}
30   {}{}
31
32 \begin{document}
33 \citeadresse{max}
34 \citeadresse{eva}
35
36 \bigskip
37 Lieber \citeanrede{max}, liebe \citeanrede{
   ↪eva},
38
39 schaut euch doch mal \cite{dante} an und
   ↪lest \cite{input1}
40
41 \printbibliography[type=adresse,title=
   ↪Verteilerliste]
42 \printbibliography[nottype=adresse,
   ↪resetnumbers]
43 \end{document}
```

Herrn
Max Muster
Im Versuchsweg 10
X01234 Testgelände

Frau
Eva Muster
Im Versuchsweg 10
X01234 Testgelände

Lieber Herr Muster, liebe Frau Muster,

schaut euch doch mal [2] an und lest [1]

Verteilerliste

[1] Max Muster, Im Versuchsweg 10, X01234 Testgelände.

[2] Eva Muster, Im Versuchsweg 10, X01234 Testgelände.

Literatur

[1] Ulrike Fischer. „Erster Text“. In: *Beispiele* (8. Apr. 2012). URL: `inputtext1.pdf`.

[2] *Internetseite dante e. V.* URL: `http://www.dante.de`.

Figure 3: The output of Listing 6

Listing 7: Unwanted cite commands

```
1 ... Namen \citeauthor{gambol} und
2 \citeauthor{dante2007} ist nicht leicht ...
3 Die göttliche Komödie ... \cite{dante2007}
```

into the “normal” bibliography. The previous examples have already shown some possibilities: `biblatex` has excellent filter features. E.g., with `nottype` one can exclude a given type from a bibliography.

But this doesn't help when we need to avoid specific cite commands leading to an entry in the bibliography. Listing 7 demonstrates the problem. It “mis-”uses the standard `\citeauthor` to facilitate the writing of a complicated name (here, from a Monty Python sketch). Neither `\citeauthor` command should trigger entries in the bibliography. But `dante2007` is cited normally later on. So it is not possible to exclude (e.g., with the keyword `skipbib`) both entries completely from the bibliography.

Listing 8 gives one possible solution to this dilemma: we introduce a new category `inbib` (line 1) and a new boolean variable `citeinbib` (lines 2–3). Using `\AtEveryCitekey`, a cited entry is added to the category if the variable is true (lines 4–6). If a cite command should not create an entry in the bibliography, `citeinbib` is set locally to false (lines 9–10), so the entry is not added to the category. Finally, `\printbibliography` can then filter using the category (line 13).

Listing 8: One solution for cite commands which should not lead to bibliography entries

```
1  \DeclareBibliographyCategory{inbib}
2  \newboolean{citeinbib}
3  \booltrue{citeinbib}
4  \AtEveryCitekey{%
5       \ifbool{citeinbib}{%
6         \addtocategory{inbib}{\thefield{
    ↪entrykey}}}{}}
7
8  \begin{document}
9  Die Schreibweise der Namen {\boolfalse{
    ↪citeinbib}\citeauthor{gambol} und
10 \citeauthor{dante2007}} ist nicht leicht zu
    ↪ merken.
11 Die göttliche Komödie ... \cite{dante2007}
12
13 \printbibliography[category=inbib]
14 \end{document}
```

Die Schreibweise der Namen Gambolputty de von Ausfern -schplenden -schlitter -crasscrenbon und Alighieri ist nicht leicht zu merken.

Die göttliche Komödie ... [1]

Literatur

[1] Dante Alighieri. *Die Göttliche Kommödie*. Übers. von Hermann Gmelin. Stuttgart, 2007.

Figure 4: The output of Listing 8

5 Summary

I hope the three examples have shown that `biblatex` offers much more than a way to print bibliographies.

References

[1] Ulrike Fischer. Eine Schnittstelle zwischen Datenbanken und LaTeX. *Die TeXnische Komödie*, 4/98:28–33, Dec. 1998.

[2] Ulrike Fischer. Serienbriefe. *Die TeXnische Komödie*, 2/99:38–44, May 1999.

[3] Herbert Voß. QR-Codes im Rand ausgeben. *Die TeXnische Komödie*, 4/13:34–37, Nov. 2013.

◇ Ulrike Fischer
Bismarckstr. 91
41061 Mönchengladbach
`fischer (at) troubleshooting-tex dot de`