

# BIBTeX Style Files for Chemistry Journals\*

Stephan Schenk  
mail (at) schenk-stephan.de

September 17, 2008

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Some notes</b>	<b>2</b>
2.1	crossref entries . . . . .	2
2.2	JAmChemSoc.bst . . . . .	3
2.3	Customization . . . . .	3
<b>3</b>	<b>Description of important entry types</b>	<b>4</b>
3.1	@article . . . . .	4
3.2	@book . . . . .	4
3.3	@inbook . . . . .	5
3.4	@incollection . . . . .	5
3.5	@masterthesis and @phdthesis . . . . .	5
3.6	@program . . . . .	6
3.7	@misc . . . . .	6
3.8	@unpublished . . . . .	6
<b>4</b>	<b>The implementation</b>	<b>7</b>
4.1	Setup . . . . .	7
4.2	Output related functions . . . . .	9
4.3	Operators . . . . .	10
4.4	Small helper functions . . . . .	11
4.5	Field formatting functions . . . . .	20
4.6	Functions related to cross-referenced entries . . . . .	26
4.7	@article related funtions . . . . .	27
4.8	@book related functions . . . . .	32
4.9	@misc related functions . . . . .	34
4.10	@program related functions . . . . .	34
4.11	Entry types . . . . .	35
4.12	Main program . . . . .	41

---

\*This file describes version 0.2.5 and has been last revised 2008/09/17.

# 1 Introduction

This collection of bibtex style files started with a version intended for *Chem. Eur. J.*. The base version of it was created using the marvelous `makebst` program by Patrick W. Daly. Soon afterwards, a version for *J. Am. Chem. Soc.* was available, too. With changes to both files becoming more and more complex, everything was merged into a single `chembst.dtx` file to avoid maintaining several versions of the code. Using the `docstrip` utility, the different style files can now easily be generated by running `latex chembst.ins`. Currently, the following journals are supported:

- *Chem. Commun.* (`ChemCommun bst`)
- *Inorg. Chem.* (`InorgChem bst`), which can be used for most of the journals published by the American Chemical Society
- *J. Am. Chem. Soc.* (`JAmChemSoc bst`)
- *Chem. Eur. J.* (`ChemEurJ bst`), which can be used for most of the journals published by Wiley

Additionally, the following styles are also supported:

- *Curriculum vitae* (`cv bst`), a style similar to *Chem. Eur. J.* that includes the title of an article

## 2 Some notes

### 2.1 crossref entries

The crossref feature is described in detail in the `btxdoc.dvi` BIBTEX documentation. Using crossref is a way to inherit information from a parent entry. As an example consider a database containing a @book entry for citing the whole book and a @inbook entry for citing some pages from that book. Naturally, the only difference between both entries is that @inbook contains a `pages` field. All the other information is stored redundantly.

Using the crossref feature the @inbook entry can be as simple as

```
@inbook{inbook_key,  
    crossref = {book_key},  
    pages = {1-5},  
}
```

All the other fields (author, publisher, year etc.) are inherited from the parent entry with the key `book_key`.

If you do not cite the whole book in your document, i. e. the document does not contain `\cite{book_key}`, the following will happen. If `book_key` is not cross-referenced more than once<sup>1</sup> everything will look perfectly like a “normal” @inbook entry. However, if `book_key` is cross-referenced more than once, the entry corresponding to the whole book will be added automatically to the bibliography and the @inbook entry will be formatted as A. Author in `\cite{book_key}`, pages 1-5.<sup>2</sup> If your document does contain `\cite{book_key}` you will always get the

---

<sup>1</sup>You can change that number by passing the argument `-min-crossrefs=number` to bibtex.

<sup>2</sup>In fact `\bibliographycite` is used. See section 2.3 for details.

crossref format.

This applies to entries of type @inbook, @incollection and @inproceedings.

## 2.2 JAmChemSoc.bst

The Journal of the American Chemical Society (ACS) now requires that for citations with more than 10 authors the list of authors is abbreviated by giving only the first author followed by *et al.* The style file `JAmChemSoc.bst` automatically takes care of that. However, ACS also requires that for those citations the full list of authors should be given as Supporting Information. The style `JAmChemSoc_all.bst` can be used for this purpose, since it does not abbreviate the list of authors.

## 2.3 Customization

Defaults for several commands used in the `thebibliography` environment are defined at the beginning of the `.bb1` file. To override the default settings define the command before using the `\bibliography` command.

`\url` The `\url` command is used to display urls. By default this only selects a typewriter font. However, it is strongly recommended, that you use the package `url.sty` which defines `\url` in a more sophisticated way. Due to some problems with multiline urls (spurious '%' in formatted url) you should use some reasonable new version (3.2 works fine) of `url.sty`.

`\urlprefix` Every `\url` is prefixed by `\urlprefix`. By default `\urlprefix` does nothing. This command is useful if you have to prepend every url with some text. For *Angew. Chem.*, i. e., every url must be preceded by "to be found under". This can easily be achieved by giving

```
\newcommand{\urlprefix}{to be found under }
```

`\foreignlanguage` This command is used to temporary switch the language for the title of books etc. By default this does nothing but if you load `babel.sty` then the language will be changed.

`\bibliographycite` This command is used for formatting crossref entries. It defaults to `\cite`. However, if you use `overcite.sty` you may not want superscripted citations in your bibliography. Then you can define `\bibliographycite` the following way

```
\newcommand{\bibliographycite}[1]{[\citen{#1}]}
```

`\bb1*` Inside the `.bb1` file no text is hard coded and instead the appropriate `\bb1*` command is used. The text "erratum", i. e., is produced by `\bb1erratum`. Thus one can easily change all the words in the bibliography by defining the appropriate commands.

```
\newcommand{\bb1erratum}{\emph{errat.}}
```

for instance will give "errat.". By redefining all necessary commands you can also change the language to some other language than the default english.

### 3 Description of important entry types

The entry types discussed in this section are those which are most commonly used in writing articles in chemistry. The types listed here refer to entries in the database (i.e. `*.bib`).

#### 3.1 @article

This entry type is designed for an article in a journal. The entry type has been augmented with several additional fields. Other style files will simply ignore these fields. The following fields are recognized:

**author** The names of all authors

**journal** The name of the journal the article was published in

**year** The year the article has been published

**volume** The volume this article has been published in

**pages** The page numbers of the article

**eid** The electronic identifier of an article. Some journals, i.e. *J. Chem. Phys.* no longer have page numbers associated to an article but only use the EID. If an EID is present **pages** will be ignored.

**numpages** The number of pages of an article. This will only be used together with an EID.

**germanpages** When citing articles from *Angew. Chem.* one should always give both the german and the english version. If this field is present the style file assumes that this is a citation of *Angew. Chem. Int. Ed.* and automatically appends the correct citation for *Angew. Chem.*. The year is always the same for both versions and the german volume is easily calculated by adding 73 to the english volume. The page range of the german version is taken from the **germanpages** field.

**erratumyear**, **erratumvolume**, **erratumpages**, **erratumeid**, **erratumnumpages**  
If any of these fields is non-empty the information will be used to append a correctly formatted citation of the erratum to the regular citation.

**note** A free format text that will be appended at the very end of a citation.

#### 3.2 @book

This entry type is designed for citing a whole book. If you want to cite only some chapters or pages of a book use `@inbook` or `@incollection` (see below). The following fields are recognized:

**author**, **editor** The names of all authors or editors. You can only use one of author or editor, but not both.

**title** The title of the book

**language** The language of the title

**edition** The edition of the book  
**volume** The volume of the book  
**series** The series the book is part of  
**publisher** The publisher of the book  
**address** The publisher's address  
**year** The year the book was published in  
**note** A free format text that will be appended at the very end of the citation.

### 3.3 @inbook

This entry type is designed for citing chapters or pages of a book. The final format of the citation is the same as for @book except that the cited chapter(s) and/or pages are appended. For the crossref feature see section 2.1. The same fields as for @book plus the following are recognized:

**chapter** The cited chapters  
**pages** The cited pages

### 3.4 @incollection

This entry is also designed for citing chapters or pages of a book and thus very similar to @inbook. However, there is one difference: the present entry is designed for chapter(s) written by some authors in a book edited by others. The final citation looks similar to A. Author, in book title, edited by E. Editor. The crossref feature is discussed in section 2.1. The same fields as for @inbook (except **title**) plus the following are recognized.

**booktitle** The name of the book

### 3.5 @masterthesis and @phdthesis

These entries are designed for citing a master's thesis or a PhD thesis. Both entries are very similar. The only difference is the thesis' name. The following fields are recognized:

**author** The author of the thesis  
**school** The school or university where this thesis work was carried out.  
**address** The school's address  
**year** The year the thesis was finished in.  
**note** A free format text that will be appended at the very end of the citation.  
**url** An url where the thesis can be found.

### **3.6 @program**

This entry is designed for citing a program. This is a new entry type which is not contained in the standard BIBTEX styles. It allows greater flexibility compared to the @misc entry. The following fields are recognized:

**author** The program's authors

**title** The program name

**description** A short description of the program

**version** The version/release of the program

**year** The year the program was published in.

**publisher** The publisher of the program

**address** The publisher's address

**note** A free format text that will be appended at the very end of the citation.

**url** An url where the program can be found.

### **3.7 @misc**

This entry is designed for everything that does not fit into one of the other entry categories. This is for instance useful to cite websites. The following fields are recognized:

**author** The author of the cited work

**title** The title of the cited work

**howpublished** A free format text describing how this work has been published.  
This is usually empty.

**year** The year the cited work was published in.

**note** A free format text that will be appended at the very end of the citation.

**url** An url where the cited work can be found.

### **3.8 @unpublished**

This entry is designed for unpublished results. The following fields are recognized:

**author** The author(s) of the unpublished work

**title** The title of the unpublished work

**year** The year the unpublished work was carried out.

**note** A free format text explaining what kind of unpublished work this is (i.e. “unpublished results”).

**url** An url where this work can be found.

## 4 The implementation

Write the journal this style file is intended for to the file header.

```
1 %% This file is intended for use with:  
2 <ChemCommun>%%          Chem. Commun.  
3 <ChemEurJ>%%            Chem.-Eur. J.  
4 <InorgChem>%%           Inorg. Chem.  
5 <JAmChemSoc>%%         J. Am. Chem. Soc.  
6 <cv>%%                 a curriculum vitae  
7 %%
```

### 4.1 Setup

ENTRY Define all the fields an entry can contain.

```
8 ENTRY  
9  {  
10   address  
11   author  
12   booktitle  
13   chapter  
14   collaboration  
15   description  
16   edition  
17   editor  
18   eid  
19   erratumeid  
20   erratumgermanpages  
21   erratumnumpages  
22   erratumpages  
23   erratumvolume  
24   erratumyear  
25   germanpages  
26   howpublished  
27   institution  
28   journal  
29   key  
30   language  
31   month  
32   note  
33   number  
34   numpages  
35   organization  
36   pages  
37   publisher  
38   school  
39   series  
40   title  
41   type  
42   url  
43   version  
44   volume  
45   year  
46 }
```

```
47  {}
48  { label }
```

INTEGERS Define all integer variables.

```
49 INTEGERS {
50   before.all
51   i
52   j
53   longest.label.width
54   mid.sentence
55   multiresult
56   nameptr
57   namesleft
58   new.sentence
59   number.label
60   numnames
61   o
62   output.state
63 }
```

STRINGS Define all string variables.

```
64 STRINGS {
65   bibinfo
66   delimiter
67   longest.label
68   s
69   t
70 }
```

bb1.\* Define all the functions for the L<sup>A</sup>T<sub>E</sub>X code which returns the words used.

```
71 FUNCTION {bb1.and} { "\bb1and{}" }
72 FUNCTION {bb1.chap} { "\bb1chap{}" }
73 FUNCTION {bb1.chapter} { "\bb1chapter{}" }
74 FUNCTION {bb1.edition} { "\bb1edn{}" }
75 FUNCTION {bb1.editor} { "\bb1ed{}" }
76 FUNCTION {bb1.editors} { "\bb1eds{}" }
77 FUNCTION {bb1.eidp} { "\bb1eidp{}" }
78 FUNCTION {bb1.eidpp} { "\bb1eidpp{}" }
79 FUNCTION {bb1.erratum} { "\bb1erratum{}" }
80 FUNCTION {bb1.etal} { "\bb1etal{}" }
81 FUNCTION {bb1.fifth} { "\bb1fiftho{}" }
82 FUNCTION {bb1.first} { "\bb1firsto{}" }
83 FUNCTION {bb1.fourth} { "\bb1fourtho{}" }
84 FUNCTION {bb1.in} { "\bb1in{}" }
85 FUNCTION {bb1.mthesis} { "\bb1mthesis{}" }
86 FUNCTION {bb1.nd} { "\bb1nd{}" }
87 FUNCTION {bb1.nr} { "\bb1no{}" }
88 FUNCTION {bb1.number} { "\bb1no{}" }
89 FUNCTION {bb1.of} { "\bb1of{}" }
90 FUNCTION {bb1.page}{ "\bb1p{}" }
91 FUNCTION {bb1.pages} { "\bb1pp{}" }
92 FUNCTION {bb1.phdthesis} { "\bb1phdthesis{}" }
93 FUNCTION {bb1.rd} { "\bb1rd{}" }
94 FUNCTION {bb1.second} { "\bb1secondo{}" }
```

```

95 FUNCTION {bb1.st} { "\bb1st{}" }
96 FUNCTION {bb1.techrep} { "\bb1techrep{}" }
97 FUNCTION {bb1.th} { "\bb1th{}" }
98 FUNCTION {bb1.third} { "\bb1third{}" }
99 FUNCTION {bb1.volume} { "\bb1vol{}" }

100 MACRO {jan} {"\bb1jan{}"}
101 MACRO {feb} {"\bb1feb{}"}
102 MACRO {mar} {"\bb1mar{}"}
103 MACRO {apr} {"\bb1apr{}"}
104 MACRO {may} {"\bb1may{}"}
105 MACRO {jun} {"\bb1jun{}"}
106 MACRO {jul} {"\bb1jul{}"}
107 MACRO {aug} {"\bb1aug{}"}
108 MACRO {sep} {"\bb1sep{}"}
109 MACRO {oct} {"\bb1oct{}"}
110 MACRO {nov} {"\bb1nov{}"}
111 MACRO {dec} {"\bb1dec{}"}

delimiter.* Define some delimiters used to separate different parts of a citation.
112 FUNCTION {delimiter.blank} { " " }
113 FUNCTION {delimiter.colon} { ":" }
114 FUNCTION {delimiter.comma} { ", " }
115 FUNCTION {delimiter.semicolon} { ";" }
116 FUNCTION {delimiter.default}
117 {
118 <ChemCommun | ChemEurJ | cv> delimiter.comma
119 <JAChemSoc | InorgChem> delimiter.semicolon
120 }

```

## 4.2 Output related functions

`output.bibitem` Write `\bibitem` and setup new citation.

```

121 FUNCTION {output.bibitem}
122 {
123   newline$ 
124   "\bibitem{" write$ 
125   cite$ write$ 
126   "}" write$ 
127   newline$ 
128   "" 
129   before.all 'output.state := 
130 }

```

`output.internal` This function is finally called by all the other output functions. It first pops the delimiter from the stack. If the (now) top string is non-empty the function appends the delimiter to the (top-1) string and writes it. The old top string is pushed back on the stack at the very end thus leaving it untouched.

```

131 FUNCTION {output.internal}
132 {
133   'delimiter := 
   write only if top string is non-empty
134   duplicate$ empty$ 
135   'pop$ 

```

```

136      {
137          's := 
138          output.state mid.sentence =
139          {
140              delimiter *
141              write$ 
142          }
143          {
144              output.state before.all =
145              'write$ 
146              { add.period$ " " * write$ }
147              if$
148              mid.sentence 'output.state :=
149          }
150          if$
151          s
152      }
153  if$
154 }

```

**output.check.internal** The function first pops the delimiter from the stack. If the (now) top string is empty it issues a warning, otherwise it calls `output.internal` for writing.

```

155 FUNCTION {output.check.internal}
156 {
157     'delimiter :=
158     't :=
159     duplicate$ empty$ 
160     { pop$ "empty " t * " in " * cite$ * warning$ }
161     { delimiter output.internal }
162     if$
163 }

```

**output** These functions just push the appropriate delimiter on the stack and then call `output.internal`.

**output.blank** 164 FUNCTION {output} { delimiter.default output.internal }

**output.semicolon** 165 FUNCTION {output.blank} { delimiter.blank output.internal }

166 FUNCTION {output.comma} { delimiter.comma output.internal }

167 FUNCTION {output.semicolon} { delimiter.semicolon output.internal }

**output.check** These functions just push the appropriate delimiter and then call `output.check.internal`.

**output.check.blank** 168 FUNCTION {output.check} { delimiter.default output.check.internal }

**output.check.comma** 169 FUNCTION {output.check.blank} { delimiter.blank output.check.internal }

**output.check.semicolon** 170 FUNCTION {output.check.comma} { delimiter.comma output.check.internal }

171 FUNCTION {output.check.semicolon} { delimiter.semicolon output.check.internal }

### 4.3 Operators

**not** Define a logical not.

```
172 FUNCTION {not}
```

```
173 {
```

```
174     { #0 }
```

```
175     { #1 }
176     if$
177 }
```

and Define a logical and.

```
178 FUNCTION {and}
179 {
180     'skip$
181     { pop$ #0 }
182     if$
183 }
```

or Define a logical or.

```
184 FUNCTION {or}
185 {
186     { pop$ #1 }
187     'skip$
188     if$
189 }
```

**multiply** Define a function for multiplying two integers.

```
190 FUNCTION {multiply}
191 {
    i is the multiplicator and will be used as a counter
192     'i :=
    j is the value to multiplicate by i, thus will be added i times to itself
193     'j :=
194     #0
195     j #0 =
    if j==0, nothing has to be done since the product will always be 0; i==0 will be
    handled gracefully by while loop below
196     'skip$
197     {
        now add j i times to the 0 on the stack
198         { i }
199         {
200             j +
201             i #1 - 'i :=
202             }
203             while$
204         }
205     if$
206 }
```

#### 4.4 Small helper functions

**bibinfo.check** This function checks whether the top value is a missing field. If so it replaces the top value by an empty string, otherwise the top value is left unchanged.

```
207 FUNCTION {bibinfo.check}
208 {
209     duplicate$ missing$
```

```

210      { pop$ "" }
211      'skip$
212      if$
213 }

```

**bibinfo.warn** This functions first checks whether a field is empty. If so it issues a warning. The topmost value is a description of the current field. If the field is missing, `bibinfo.warn` pushes an empty string, otherwise the value is left unchanged. The behaviour is very similar to `bibinfo.check`.

```

214 FUNCTION {bibinfo.warn}
215 {
216   swap$
217   duplicate$ missing$
218   {
219     swap$ "missing " swap$ * " in " * cite$ * warning$ pop$
220     ""
221   }
222   { duplicate$ empty$
223   {
224     swap$ "empty " swap$ * " in " * cite$ * warning$
225   }
226   { swap$
227     pop$
228   }
229   if$
230 }
231 if$
232 }

```

**bolden** This function returns the L<sup>A</sup>T<sub>E</sub>X code for boldening the top string.

```

233 FUNCTION {bolden}
234 {
235   duplicate$ empty$
236   { pop$ "" }
237   { "\textbf{" swap$ * "}" * }
238   if$
239 }

```

**capitalize** This function capitalizes the first letter of a word.

```

240 FUNCTION {capitalize}
241 {
242   "\capitalise" swap$ *
243 }

```

**cat.internal** This function catenates two strings using the delimiter on top of the stack. The second string is at (top-1) position, the first at (top-2). If any of both strings is empty the function just returns the other string without any delimiter.

```

244 FUNCTION {cat.internal}
245 {
246   'delimiter :=
247   duplicate$ empty$
248   'pop$
249   {

```

```

250      swap$  

251      duplicate$ empty$  

252          'skip$  

253          { delimiter * }  

254      if$  

255      swap$  

256      *  

257  }  

258  if$  

259 }

cat.blank These functions just push the appropriate delimiter and call cat.internal.  

cat.colon 260 FUNCTION {cat.blank} { delimiter.blank cat.internal }  

cat.comma 261 FUNCTION {cat.colon} { delimiter.colon cat.internal }  

cat.default 262 FUNCTION {cat.comma} { delimiter.comma cat.internal }  

cat.semicolon 263 FUNCTION {cat.default} { delimiter.default cat.internal }  

              264 FUNCTION {cat.semicolon} { delimiter.semicolon cat.internal }

```

**eng.ord** This function formats an english ordinal by appending the appropriate string.

```

265 FUNCTION {eng.ord}  

266 {  

267     duplicate$ "1" swap$ *  

268     #-2 #1 substring$ "1" =  

269         { bbl.th * }  

270         { duplicate$ #-1 #1 substring$  

271             duplicate$ "1" =  

272                 { pop$ bbl.st * }  

273                 { duplicate$ "2" =  

274                     { pop$ bbl.nd * }  

275                     { "3" =  

276                         { bbl.rd * }  

277                         { bbl.th * }  

278                     if$  

279                 }  

280             if$  

281         }  

282     if$  

283     }  

284     if$  

285 }

```

**is.num** This function checks whether the top character is a number.

```

286 FUNCTION {is.num}  

287 {  

288     chr.to.int$  

289     duplicate$ "0" chr.to.int$ < not  

290     swap$ "9" chr.to.int$ > not and  

291 }

```

**extract.num** This function extracts a number (as string) from the top string.

```

292 FUNCTION {extract.num}  

293 {  

294     duplicate$ 't :=  

295     "" 's :=

```

```

296 { t empty$ not }
297 {
298   t #1 #1 substring$
299   t #2 global.max$ substring$ 't := 
300   duplicate$ is.num
301   { s swap$ * 's := }
302   { pop$ "" 't := }
303   if$
304 }
305 while$
306 s empty$
307   'skip$
308   { pop$ s }
309 if$
310 }

```

`convert.edition` This functions converts the `edition` field into the appropriate ordinal.

```

311 FUNCTION {convert.edition}
312 {
313   extract.num "1" change.case$ 's :=
314   s "first" = s "1" = or
315   { bbl.first 't := }
316   { s "second" = s "2" = or
317     { bbl.second 't := }
318     { s "third" = s "3" = or
319       { bbl.third 't := }
320       { s "fourth" = s "4" = or
321         { bbl.fourth 't := }
322         { s "fifth" = s "5" = or
323           { bbl.fifth 't := }
324           { s #1 #1 substring$ is.num
325             { s eng.ord 't := }
326             { edition 't := }
327             if$
328           }
329           if$
330         }
331         if$
332       }
333       if$
334     }
335     if$
336   }
337   if$
338   t
339 }

```

`either.or.check` This function checks whether mutually exclusive fields are present at the same time.

```

340 FUNCTION {either.or.check}
341 {
342   empty$
343   'pop$
344   { "can't use both " swap$ * " fields in " * cite$ * warning$ }

```

```
345   if$  
346 }
```

**emphasize** This function returns the L<sup>A</sup>T<sub>E</sub>X code for emphasizing the top string.

```
347 FUNCTION {emphasize}  
348 {  
349   duplicate$ empty$  
350   { pop$ "" }  
351   { "\emph{" swap$ * "}" * }  
352   if$  
353 }
```

**fin.entry** This function finalizes a citation. It appends a dot and writes the last text chunk.

```
354 FUNCTION {fin.entry}  
355 {  
356   add.period$  
357   write$ newline$  
358 }
```

**format.names** This function formats a list of names.

```
359 FUNCTION {format.names}  
360 {  
361 % bibinfo is the description of the names, i.e. author, editor  
362   'bibinfo :=  
363   duplicate$ empty$  
364   'skip$  
365   {  
366     s is the full list of names  
367     's :=  
368     t is the formatted name  
369     "" 't :=  
370     nameptr is the index of the current name  
371     numnames is the total number of names  
372     namesleft is the number of names yet to format  
373     #1 'nameptr :=  
374     s num.names$ 'numnames :=  
375     numnames 'namesleft :=  
376     { namesleft #0 > }  
377     {  
378       s nameptr  
379       {ChemCommun | ChemEurJ | cv}      "{f.~}{vv~}{11}{, jj}"  
380       {JAICheMSoC | InorgChem}        "{vv~}{11}{, f.}{, jj}"  
381       format.name$  
382       bibinfo.check  
383       't :=  
384       nameptr #1 >  
385       {
```

The following code chunk checks whether the list of names should be abbreviated by et al.

```
386   numnames #0  
387   {tennames}      #10 +
```

```

383 <fifteennames>           #15 +
384      >
If no appropriate docstrip option is given, the next expression will always be
false (nameptr is always  $> 1$ , see above). Thus the check is effectively disabled.
385      nameptr #0
386 <namesone>           #2 +
387      =
388      and
389      {
list of names should be truncated, set formatted name to "others"
390          "others" 't :=
391          #1 'namesleft :=
392          }
393          'skip$
394          if$
395          delimiter.default *
396          namesleft #1 >
397          { t * }
398          {
check whether current name is "others", if so set formatted name to "others"
399          s nameptr "{11}" format.name$
400          duplicate$ "others" =
401          { 't := }
402          { pop$ }
403          iff
check whether formatted name is "others", if so print et al.
404          t "others" =
405          { bbl.etal * }
406          { t * }
407          iff
408          }
409          iff
410          }
this is the first name
411          't
412          iff
413          nameptr #1 + 'nameptr :=
414          namesleft #1 - 'namesleft :=
415          }
416          while$
417          }
418          iff
419 }

get.bbl.editor Return "editors" if editor contains more than one name, "editor" otherwise.
420 FUNCTION {get.bbl.editor}
421 {
422   editor num.names$ #1 >
423   'bbl.editors
424   'bbl.editor
425   iff
426 }

```

`get.bbl.erratum` Return the formatted string for “erratum”.

```
427 FUNCTION {get.bbl.erratum}
428 {
429   bbl.erratum
430   ":" *
431 }
```

`multi.page.check` Check whether the top string is only a single page or a range of pages.

```
432 FUNCTION {multi.page.check}
433 {
434   't :=
435   #0 'multiresult :=
436   {
437     multiresult not
438     t empty$ not and
439   }
440   { t #1 #1 substring$
441     duplicate$ "-" =
442       swap$ duplicate$ "," =
443       swap$ "+" =
444       or or
445       { #1 'multiresult := }
446       { t #2 global.max$ substring$ 't := }
447     if$
448   }
449   while$
450   multiresult
451 }
```

`get.bbl.page` Return a formatted prefix for pages.

```
452 FUNCTION {get.bbl.page}
453 {
454   duplicate$ multi.page.check
455   { bbl.pages }
456   { bbl.page }
457   if$
458 {ChemCommun | ChemEurJ | cv}  "." *
459 }
```

`n.dashify` Replace a single “-” in a range of pages by “--”.

```
460 FUNCTION {n.dashify}
461 {
462   't :=
463   ""
464   { t empty$ not }
465   {
466     t #1 #1 substring$ "-" =
467     {
468       t #1 #2 substring$ "--" = not
469       {
470         "--" *
471         t #2 global.max$ substring$ 't :=
472       }
473     }
```

there is more than one '−', therefore append all of them to the string on the stack

```
474          { t #1 #1 substring$ "−" = }
475          {
476          "−" *
477          t #2 global.max$ substring$ 't :=
478          }
479          while$
480          }
481          if$
482          }
483          {
```

the current char is not '−', therefore just append it

```
484          t #1 #1 substring$ *
485          t #2 global.max$ substring$ 't :=
486          }
487          if$
488          }
489          while$
490 }
```

**select.language** Return L<sup>A</sup>T<sub>E</sub>X code for changing the language for the string on top of the stack.

```
491 FUNCTION {select.language}
492 {
493   duplicate$ empty$
494   'skip$
495   {
496     language empty$
497     'skip$
498     { "\foreignlanguage{" language * "}{ " * swap$ * "}" * }
499   if$
500   }
501   if$
502 }
```

**space.word** This function puts spaces around a word.

```
503 FUNCTION {space.word}
504 {
505   " " swap$ * " " *
506 }
```

**str.to.int.warn** Print a warning if top string is not a representation of a valid integer (used by **str.to.int**).

```
507 FUNCTION {str.to.int.warn}
508 {
509   "str.to.int: '' swap$ * '' is not a valid integer" * warning$
510 }
```

**str.to.int** This function converts a string into an integer. A warning is issued if the string is not a valid representation of an integer.

```
511 FUNCTION {str.to.int}
512 {
513   duplicate$ empty$
514   {
```

```

515      str.to.int.warn
516      #0
517    }
518  {
assign the original string to t for parsing t from the end
519      duplicate$ 't :=
check for sign
520      t #1 #1 substring$ "-" =
521      {
be sure that "-" is followed by at least one more character
522          t #2 global.max$ substring$ 't :=
523          t empty$ 
524          {
525              duplicate$ str.to.int.warn
526              #0
527          }
528          { #-1 }
529          if$
530      }
531      { #1 }
532      if$

the top stack position contains now -1 or 1 depending on sign
o stores the offset for position inside the number
533      #1 'o :=
push starting value on stack
534      #0
535      { t empty$ not}
536      {
get last character
537      t #-1 #1 substring$ 
538      duplicate$ is.num
539      {
character is in range [0-9], now multiply by offset and add to value already on
stack
540          chr.to.int$ #48 -
541          o multiply
542          +
remove last character from string, increment offset o
543          t #-2 global.max$ substring$ 't :=
544          o #10 multiply 'o :=
545      }
546      {
the last character was not a digit, therefore pop last character and sum
547          pop$ pop$
swap sign and original string, duplicate string, print warning
548          swap$ duplicate$ str.to.int.warn
swap original string and sign, push 0
549          swap$ #0

```

```

break while loop
550      "" 't := "
551      }
552      if$
553      }
554      while$  

stack holds value and sign, multiply combines them
555      multiply
pop copy of original string
556      swap$ pop$  

557      }
558      if$  

559 }

tie.or.space.prefix This function prepends a string with either a space or a “~” depending on the length of the string.
560 FUNCTION {tie.or.space.prefix}
561 {
562   duplicate$ text.length$ #3 <
563   { "~" }
564   { " " }
565   if$  

566   swap$  

567 }

word.in This function returns the word “in” followed by a space.
568 FUNCTION {word.in}
569 {
570   bbl.in delimiter.blank *
571 }

```

## 4.5 Field formatting functions

```

format.year.internal This function applies all necessary formatting for a year on the stack.
572 FUNCTION {format.year.internal}
573 {
574   ⟨ChemEurJ | cv⟩ bolden
575 }

format.volume.internal This function applies all necessary formating for a volume on the stack.
576 FUNCTION {format.volume.internal}
577 {
578   duplicate$ empty$  

579   'skip$  

580   {
581     bbl.volume
582     swap$  

583     tie.or.space.prefix
stack (top-down): volume, tie/space, bbl.volume
584     * *
585   }

```

```
586 if$  
587 }
```

**format.authors** This function formats the list of authors. If a `collaboration` field is present its contents is printed and the list of authors is appended in parentheses.

```
588 FUNCTION {format.authors}  
589 {  
590   author "author" format.names  
591   duplicate$ empty$  
592     'skip$  
593   {  
594     collaboration bibinfo.check  
595     duplicate$ empty$  
596       'skip$  
597       { " (" * swap$ * ")" * }  
598     if$  
599     *  
600   }  
601   if$  
602 }
```

**format.booktitle** This function formats the title of a book and switches to an optional foreign language for this title.

```
603 FUNCTION {format.booktitle}  
604 {  
605   booktitle bibinfo.check  
606   emphasize  
607   select.language  
608 }
```

**format.chapter** This function formats a chapter. If a `type` field is present its contents is used as a prefix. If it is absent then the default (return value of `bbl.chapter`) will be used.

```
609 FUNCTION {format.chapter}  
610 {  
611   chapter bibinfo.check  
612   duplicate$ empty$  
613     'skip$  
614   {  
615     type bibinfo.check  
616     duplicate$ empty$  
617       { pop$ bbl.chapter }  
618       { "1" change.case$ }  
619     if$  
620   <JAmChemSoc | InorgChem>      capitalize  
621     swap$  
622     stack (top-down): chapter, type/bbl.chapter  
623       tie.or.space.prefix  
624     * *  
625   }  
626   if$  
627 }
```

**format.date** This function formats the date of a citation. All currently supported styles only use the year. It is pushed onto the stack and `format.year.internal` is called to format it.

```
627 FUNCTION {format.date}
628 {
  629   year bibinfo.check
  630   format.year.internal
  631 }
```

**format.edition** This function formats the edition. It is converted into the appropriate english ordinal and the return value of `bbl.edition` will be appended.

```
632 FUNCTION {format.edition}
633 {
  634   edition bibinfo.check
  635   duplicate$ empty$
  636   'skip$
  637   {
    638     convert.edition
    639     output.state mid.sentence =
    640     { "1" }
    641     { "t" }
    642     if$ change.case$
    643     " " * bbl.edition *
  644 <ChemCommun | ChemEurJ | cv>      emphasize
  645   }
  646   if$
  647 }
```

**format.editors** This function formats the list of editors.

```
648 FUNCTION {format.editors}
649 {
  650   editor "editor" format.names
  651   duplicate$ empty$
  652   'skip$
  653   {
    654 <JAmChemSoc | InorgChem>      author empty$
    655 <JAmChemSoc | InorgChem>      'skip$
    656 <JAmChemSoc | InorgChem>      {
    657 <JAmChemSoc | InorgChem>      ","
    658     " " *
    659     get.bbl.editor
    660     capitalize
    661 <ChemCommun | ChemEurJ | cv>    "(" swap$ * ")" *
    662     *
    663 <JAmChemSoc | InorgChem>      }
    664 <JAmChemSoc | InorgChem>      if$
    665   }
    666   if$
  667 }
```

**format.in.booktitle** This function formats a booktitle an prepends it with “in”.

```
668 FUNCTION {format.in.booktitle}
669 {
```

```

670   format.booktitle
671   duplicate$ empty$
672     'skip$
673   {
674     word.in
675   <JAmChemSoc | InorgChem>      capitalize
676     swap$ *
677   }
678   if$
679 }

```

**format.note** This function formats the `note` field of a citation. Since this is a free format field is content is used unchanged.

```

680 FUNCTION {format.note}
681 {
682   note bibinfo.check
683 }

```

**format.number.series** This function formats number and series if the `volume` field is empty.

```

684 FUNCTION {format.number.series}
685 {
686   volume bibinfo.check
687   duplicate$ empty$
688   {
689     number empty$
690     { series bibinfo.check }
691   {
692     series empty$
693     { number bibinfo.check }
694   {
695     output.state mid.sentence =
696     { bbl.number }
697     { bbl.number capitalize }
698     if$
699     number bibinfo.check tie.or.space.prefix * *
700     word.in *
701     series bibinfo.check *
702   }
703   if$
704   }
705   if$
706   }
707   'skip$
708   if$
709 }

```

**format.org.or.pub** This function formats an organization or publisher (on the stack) and its address.

```

710 FUNCTION {format.org.or.pub}
711 {
712   't :=
713   address empty$ t empty$ and
714   { "" }
715   {
716     t

```

```

717     address bibinfo.check
718     duplicate$ empty$
719         'pop$
720         {
721 <ChemCommun | ChemEurJ | cv>           cat.comma
722 <JAmChemSoc | InorgChem>             cat.colon
723         }
724         if$
725     }
726     if$
727 }

```

**format.organization.address** This function formats an organization and its address. It pushes the organization onto the stack and calls `format.org.or.pub`.

```

728 FUNCTION {format.organization.address}
729 {
730   organization bibinfo.check
731   format.org.or.pub
732 }

```

**format.pages** This function formats a list of pages. The list is prepended with `bbl.page(s)`.

```

733 FUNCTION {format.pages}
734 {
735   pages bibinfo.check
736   duplicate$ empty$
737       'skip$
738       {
739         n.dashify
740         get.bbl.page
741         stack top-down: pages prefix, pages
742         swap$
743         tie.or.space.prefix *
744     }
745     if$
746 }

```

**format.publisher.address** This function formats a publisher and its address. It pushes the publisher onto the stack and calls `format.org.or.pub`.

```

747 FUNCTION {format.publisher.address}
748 {
749   publisher "publisher" bibinfo.warn
750   format.org.or.pub
751 }

```

**format.thesis.type** This function formats the type of a thesis.

```

752 FUNCTION {format.thesis.type}
753 {
754   type
755   duplicate$ empty$
756   'pop$
757   {
758     swap$ pop$

```

```

759     "t" change.case$ bibinfo.check
760   }
761   if$
762 }

format.title This function formats a title. It also switches the language temporarily.
763 FUNCTION {format.title}
764 {
765   title bibinfo.check
766   duplicate$ empty$
767   'skip$
768   {
769     emphasize
770     select.language
771   }
772   if$
773 }

format.tr.number This function formats the number of a technical report.
774 FUNCTION {format.tr.number}
775 {
776   number bibinfo.check
777   type
778   duplicate$ empty$
779   { pop$ bbl.techrep }
780   'skip$
781   if$
782   bibinfo.check
783   swap$
784   duplicate$ empty$
785   { pop$ "t" change.case$ }
786   { tie.or.space.prefix * * }
787   if$
788 }

format.url This function formats an url. Each url is prefixed with \urlprefix. See section
2.3 for more information.
789 FUNCTION {format.url}
790 {
791   url bibinfo.check
792   duplicate$ empty$
793   'skip$
794   {
795     "\urlprefix"url{" swap$ * "}" *
796     new.sentence 'output.state :=
797   }
798   if$
799 }

format.volume This function formats the volume field. It merely pushes the field's value onto the
stack and calls format.volume.internal to format it.
800 FUNCTION {format.volume}
801 {

```

```
802   volume bibinfo.check  
803   format.volume.internal  
804 }
```

`format.volume.and.series` This function will format the `volume` and `series`, but only if *both* of them are not empty. Otherwise an empty string is returned.

```
805 FUNCTION {format.volume.and.series}  
806 {  
807   volume empty$ series empty$ or  
808   { "" }  
809   {  
810     volume format.volume.internal  
811     prepend "of"  
812     swap$ bbl.of space.word * swap$  
813     emphasize  
814     *  
815     "volume and number" number either.or.check  
816   }  
817   if$  
817 }
```

`format.volume.noseries` This function formats the volume. If the `series` field is non-empty an empty string is returned.

```
818 FUNCTION {format.volume.noseries}  
819 {  
820   series empty$  
821   {  
822     volume bibinfo.check  
823     format.volume.internal  
824 <ChemCommun | ChemEurJ | cv>      emphasize  
825   }  
826   { "" }  
827   if$  
828 }
```

## 4.6 Functions related to cross-referenced entries

`bibliography.cite` This function returns the string for citing the document the `crossref` field points to.

```
829 FUNCTION {bibliography.cite}  
830 {  
831   "\bibliographycite{" swap$ * "}" *  
832 }
```

`format.crossref` This function formats the cross-reference.

```
833 FUNCTION {format.crossref}  
834 {  
835   bbl.in " " *  
836 <JAmChemSoc | InorgChem>  capitalize  
837   crossref bibliography.cite *  
838 }
```

## 4.7 @article related funtions

rmat.article.cat.journal.year	This function catenates the journal name and the year. It is used when formatting an erratum and/or articles in <i>Angew. Chem.</i>
	839 FUNCTION {format.article.cat.journal.year} 840 { 841 <ChemCommun> cat.comma 842 <ChemEurJ   cv   JAmChemSoc   InorgChem> cat.blank 843 }
format.article.year.internal	This function provides all formatting required for an article which is not already done in format.date.internal.
	844 FUNCTION {format.article.year.internal} 845 { 846 <JAmChemSoc   InorgChem> bolden 847 }
format.article.volume.internal	This function applies all necessary formatting to the volume an article is published in.
	848 FUNCTION {format.article.volume.internal} 849 { 850 <ChemCommun> bolden 851 <ChemEurJ   cv   JAmChemSoc   InorgChem> emphasize 852 }
e.germanpages.volume.internal	This function is used while formatting citations of <i>Angew. Chem. Int. Ed.</i> . It calculates the volume of the german edition from the volume of the english one (on the stack).
	853 FUNCTION {format.article.germanpages.volume.internal} 854 { 855   duplicate\$ empty\$ 856     'skip\$ 857   { 858     str.to.int 859     duplicate\$ #1 < 860       { 861         pop\$ 862         "volume in \" cite\$ * \" is not a positive integer value\" * warning\$ 863         "" 864       } 865     { vol. 1 of the english edition corresponds to vol. 74 of the german one 866           #73 + 867           int.to.str\$ 868           format.article.volume.internal 869         } 870       if\$ 871     } 872   if\$ 873 }
le.germanpages.pages.internal	This function applies all necessary formatting to pages of <i>Angew. Chem.</i> , the german edition of <i>Angew. Chem. Engl. Ed.</i>

```

874 FUNCTION {format.article.germanpages.pages.internal}
875 {
876   n.dashify
877 }

format.article.date This function formats the date of an article.
878 FUNCTION {format.article.date}
879 {
880   format.date
881   format.article.year.internal
882 }

format.article.numpages This function formats the number of pages of an article. The number of pages
(numpages field) is only used in conjunction with an EID. The number of pages is
assumed to be already on the stack.
883 FUNCTION {format.article.numpages}
884 {
885   duplicate$ empty$
886   'skip$
887   {
888     duplicate$ "1" =
889     { "~" * bbl.eidp * }
890     { "~" * bbl.eidpp * }
891     if$
892     "(" swap$ * ")" *
893   }
894   if$
895 }

format.article.eid This function formats the EID of an article.
896 FUNCTION {format.article.eid}
897 {
898   eid bibinfo.check
899   duplicate$ empty$
900   'pop$
901   {
902     cat.comma
903   <cv>      numpages bibinfo.check
904   <cv>      format.article.numpages
905   <cv>      cat.blank
906   }
907   if$
908 }

format.article.journal This function formats the journal an article was published in.
909 FUNCTION {format.article.journal}
910 {
911   journal bibinfo.check
912   duplicate$ empty$
913   'skip$
914   { emphasize }
915   if$
916 }

```

ticle.germanpages.journalname This function formats *Angew. Chem.* as the name of the german version of *Angew. Chem. Int. Ed.*.

```
917 FUNCTION {format.article.germanpages.journalname}
918 {
919   "Angew.\ Chem."
920   emphasize
921 }
```

cle.erratum.germanpages.pages This function is used for an erratum in *Angew. Chem. Int. Ed.*. It formats the pages of the german version.

```
922 FUNCTION {format.article.erratum.germanpages.pages}
923 {
924   erratumgermanpages bibinfo.check
925   format.article.germanpages.pages.internal
926 }
```

format.article.erratum.year This function formats the year an erratum was published.

```
927 FUNCTION {format.article.erratum.year}
928 {
929   erratumyear bibinfo.check
930   format.year.internal
931   format.article.year.internal
932 }
```

le.erratum.germanpages.volume This function is used for an erratum in *Angew. Chem. Int. Ed.*. It formats the erratum's volume in the german version.

```
933 FUNCTION {format.article.erratum.germanpages.volume}
934 {
935   erratumvolume "erratumvolume" bibinfo.warn
936   format.article.germanpages.volume.internal
937 }
```

t.article.erratum.germanpages This function formats the german version of an erratum published in *Angew. Chem. Int. Ed.*.

```
938 FUNCTION {format.article.erratum.germanpages}
939 {
940   erratumgermanpages empty$
941   { ""
942   {
943     format.article.germanpages.journalname
944     format.article.erratum.year format.article.cat.journal.year
945     format.article.erratum.germanpages.volume cat.comma
946     format.article.erratum.germanpages.pages cat.comma
947   }
948   if$
949 }
```

format.article.erratum.journal This function formats the journal of an erratum. Normally this is the same journal the original article was published in.

```
950 FUNCTION {format.article.erratum.journal} { format.article.journal }
```

format.article.erratum.pages This function formats the pages of an erratum.

```
951 FUNCTION {format.article.erratum.pages}
```

```

952 {
953   erratumpages bibinfo.check
954   n.dashify
955 }

format.article.erratum.eid This function formats the EID of an erratum.
956 FUNCTION {format.article.erratum.eid}
957 {
958   erratumeid bibinfo.check
959   erratumnumpages bibinfo.check format.article.numpages *
960 }

format.article.erratum.volume This function formats the volume of an erratum.
961 FUNCTION {format.article.erratum.volume}
962 {
963   erratumvolume bibinfo.check
964   format.article.volume.internal
965 }

format.article.erratum This function formats an erratum of an article. The erratum will be formatted if
one of erratumyear, erratumvolume, erratumpages or erratumeid is present.
966 FUNCTION {format.article.erratum}
967 {
968   erratumyear bibinfo.check empty$*
969   erratumvolume bibinfo.check empty$ and
970   erratumpages bibinfo.check empty$ and
971   erratumeid bibinfo.check empty$ and
972   { "" }
973   {

An erratum was detected. Since at least one field is present, “erratum:” can
already be printed.
974   get.bbl.erratum
975   format.article.erratum.journal cat.blank
976   format.article.erratum.year format.article.cat.journal.year
977   format.article.erratum.volume cat.comma
978   erratumeid empty$*
979   { format.article.erratum.pages }
980   { format.article.erratum.eid }
981   if$*
982   cat.comma
983   format.article.erratum.germanpages cat.semicolon
984   }
985   if$*
986 }

mat.article.germanpages.pages This function formats the pages of the german version of an article in Angew. Chem. Int. Ed.
987 FUNCTION {format.article.germanpages.pages}
988 {
989   germanpages bibinfo.check
990   format.article.germanpages.pages.internal
991 }

```

**at.article.germanpages.volume** This function formats the volume of the german version of an article in *Angew. Chem. Int. Ed.*. It pushes the volume of the english version onto the stack and calls `format.article.germanpages.volume.internal` which automatically calculates and formats the german volume.

```

992 FUNCTION {format.article.germanpages.volume}
993 {
994   volume "volume" bibinfo.warn
995   format.article.germanpages.volume.internal
996 }
    
```

**format.article.germanpages.year** This function formats the year of the german version of an article in *Angew. Chem. Int. Ed.*. This is always the same as the english version.

```

997 FUNCTION {format.article.germanpages.year} { format.article.date }
    
```

**format.article.germanpages** This function formats the german version of an article in *Angew. Chem. Int. Ed.*.

```

998 FUNCTION {format.article.germanpages}
999 {
1000   germanpages empty$ 
1001   { "" }
1002   {
1003     format.article.germanpages.journalname
1004     format.article.germanpages.year format.article.cat.journal.year
1005     format.article.germanpages.volume cat.comma
1006     format.article.germanpages.pages cat.comma
1007   }
1008   if$
1009 }
    
```

**format.article.pages** This function formats the pages of an article.

```

1010 FUNCTION {format.article.pages}
1011 {
    
```

The stack contains whatever is the text directly preceding pages.

```

1012   pages
1013   duplicate$ empty$ 
1014   'pop$ 
1015   {
    
```

Check whether preceding string is empty.

```

1016   swap$ 
1017   duplicate$ empty$ 
    
```

If the preceding string is empty, pop it and the pages from stack and call `format.pages`. This means that `pages` is the only text so far (except bibitem).

```

1018   { pop$ pop$ format.pages }
    
```

The preceding string is not empty. Therefore format the pages and append them.

```

1019   {
1020     swap$ 
1021     n.dashify
1022     cat.comma
1023   }
1024   if$
1025   }
1026   if$
1027 }
    
```

`format.article.title` This function formats the title of an article.

```
1028 FUNCTION {format.article.title}
1029 {
1030   title bibinfo.check
1031   duplicate$ empty$
1032     'skip$
1033   {
1034     new.sentence 'output.state :=
1035   }
1036   if$
1037 }
```

`mat.article.volume.and.number` This function formats the volume an article was published in. Optionally it appends the issue to the volume.

```
1038 FUNCTION {format.article.volume.and.number}
1039 {
1040   volume bibinfo.check
1041   duplicate$ empty$
1042     'skip$
1043   { bibinfo.check }
1044   if$
1045   format.article.volume.internal
1046 (*number)
1047   number bibinfo.check
1048   duplicate$ empty$
1049     'skip$
1050   {
1051     swap$
1052     duplicate$ empty$
1053       { "there's a number but no volume in " cite$ * warning$ }
1054     'skip$
1055   if$
1056   swap$
1057   "(" swap$ * ")" *
1058 }
1059   if$
1060   *
1061 (/number)
1062 }
```

## 4.8 @book related functions

`format.book.authors` This function formats the authors of a book. If no authors were given, the editors are used instead.

```
1063 FUNCTION {format.book.authors}
1064 {
1065   author empty$
1066   { format.editors }
1067   {
1068     format.authors
1069     "author and editor" editor either.or.check
1070   }
1071   if$
```

```
1072 }
```

`format.book.editors` This function formats the editors of a book. If the book has no authors then the editors were already given as a substitute for the authors. In this case, nothing is printed, since the editors should not be given twice.

```
1073 FUNCTION {format.book.editors}
1074 {
1075   author empty$ 
1076   { "" }
1077   { format.editors }
1078   if$
1079 }
```

`format.book.volume.internal` This function formats the volume of a book.

```
1080 FUNCTION {format.book.volume.internal}
1081 {
1082   volume bibinfo.check
1083   duplicate$ empty$ 
1084   'skip$ 
1085   {
1086     tie.or.space.prefix *
1087     bbl.volume swap$ *
1088   }
1089   if$
1090 }
```

`format.book.volume.and.series` This function formats the volume of a book and the series it is part of. For this function to produce any non-empty output both `volume` and `series` must be non-empty. To format the volume without a series the `format.book.noseries` function is used (see below).

```
1091 FUNCTION {format.book.volume.and.series}
1092 {
1093   series bibinfo.check
1094   duplicate$ empty$ 
1095   'skip$ 
1096   {
1097   <ChemCommun | ChemEurJ | cv>      emphasize
```

Now the formatted series is on the stack.

```
1098       format.book.volume.internal
1099   <ChemCommun | ChemEurJ | cv>      bbl.of space.word *
1100   stack top-down: formatted volume; formatted series
1101   <ChemCommun | ChemEurJ | cv>      swap$ *
1102   <JAChemSoc | InorgChem>          cat.comma
1103   if$
1104 }
```

`format.book.volume.noseries` This function formats the volume of a book if the `series` field is empty.

```
1105 FUNCTION {format.book.volume.noseries}
1106 {
1107   series empty$ not
1108   { "" }
```

```

1109      {
1110        format.book.volume.internal
1111 <ChemCommun | ChemEurJ | cv>      emphasize
1112      }
1113    if$
1114 }

```

## 4.9 @misc related functions

`misc.empty.check` This function checks whether fields of an @misc entry are empty and issues a warning.

```

1115 FUNCTION {misc.empty.check}
1116 {
1117   author empty$ title empty$ howpublished empty$
1118   month empty$ year empty$ note empty$ url empty$
1119   and and and and and
1120   { "all relevant fields are empty in " cite$ * warning$ }
1121   'skip$
1122   if$
1123 }

```

## 4.10 @program related functions

`format.program.description` This function formats the description of a program. It also temporarily switches the language.

```

1124 FUNCTION {format.program.description}
1125 {
1126   description bibinfo.check
1127   duplicate$ empty$
1128   'skip$
1129   { select.language }
1130   if$
1131 }

```

`format.program.publisher.address` This function formats the publisher of a program and its address.

```

1132 FUNCTION {format.program.publisher.address}
1133 {
1134   publisher bibinfo.check
1135   format.org.or.pub
1136 }

```

`format.program.title` This function formats the title of a program.

```

1137 FUNCTION {format.program.title}
1138 {
1139   title "title" bibinfo.warn
1140   duplicate$ empty$
1141   'skip$
1142   {
1143     "t" change.case$
1144     " " swap$ *
1145     capitalize
1146     emphasize
1147   }

```

```
1148 if$  
1149 }
```

`format.program.version` This function formats the version of a program.

```
1150 FUNCTION {format.program.version}  
1151 {  
1152   version  
1153 }
```

## 4.11 Entry types

`article`

```
1154 FUNCTION {article}  
1155 {  
1156   output.bibitem  
1157   format.authors "author"  
1158 <ChemCommun | ChemEurJ | cv> output.check  
1159 <JAmChemSoc | InorgChem> output.check.blank  
1160 <cv> format.article.title "title" output.check  
1161 <cv> new.sentence 'output.state :=  
1162   format.article.journal "journal"  
1163 <ChemCommun | ChemEurJ | cv> output.check  
1164 <JAmChemSoc | InorgChem> output.check.blank  
1165   format.article.date "year"
```

If you change the way journal name and publication year are catenated, do not forget to change `format.article.cat.journal.year`.

```
1166 <ChemCommun> output.check  
1167 <ChemEurJ | cv | JAmChemSoc | InorgChem> output.check.blank  
1168   format.article.volume.and.number output.comma  
1169   eid empty$  
1170     { format.article.pages }  
1171     { format.article.eid }  
1172   if$  
1173     format.article.germanpages output.semicolon  
1174     format.article.erratum output.semicolon  
1175     format.note output  
1176   fin.entry  
1177 }
```

`book`

```
1178 FUNCTION {book}  
1179 {  
1180   output.bibitem  
1181   format.book.authors "author and editor" output.check  
1182   format.title "title"  
1183 <ChemCommun | ChemEurJ | cv> output.check.comma  
1184 <JAmChemSoc | InorgChem> output.check.blank  
1185   format.edition  
1186 <ChemCommun | ChemEurJ | cv> output.blank  
1187 <JAmChemSoc | InorgChem> output.comma  
1188 <JAmChemSoc | InorgChem> format.book.editors output.semicolon  
1189   format.book.volume.and.series output  
1190 <ChemCommun | ChemEurJ | cv> format.book.volume.noseries output
```

```

1191 <ChemCommun | ChemEurJ | cv> format.book.editors output.comma
1192   format.publisher.address output
1193   format.date "year" output.check.comma
1194 <JAmChemSoc | InorgChem> format.volume.noseries output.semicolon
1195   format.note output
1196   fin.entry
1197 }

booklet
1198 FUNCTION {booklet}
1199 {
1200   output.bibitem
1201   format.authors output
1202   format.title "title" output.check
1203   howpublished bibinfo.check output
1204   address bibinfo.check output
1205   format.date output
1206   format.note output
1207   fin.entry
1208 }

inbook
1209 FUNCTION {inbook}
1210 {
1211   output.bibitem
1212   format.book.authors "author and editor" output.check
1213   crossref missing$
1214   {
1215     format.title "title"
1216   <ChemCommun | ChemEurJ | cv> output.check.comma
1217   <JAmChemSoc | InorgChem> output.check.blank
1218   format.edition
1219   <ChemCommun | ChemEurJ | cv> output.blank
1220   <JAmChemSoc | InorgChem> output.comma
1221   <JAmChemSoc | InorgChem> format.book.editors output.semicolon
1222   format.book.volume.and.series output
1223   <ChemCommun | ChemEurJ | cv> format.book.volume.noseries output
1224   <ChemCommun | ChemEurJ | cv> format.book.editors output.comma
1225   format.publisher.address output
1226   format.date "year" output.check.comma

The J. Am. Chem. Soc. requires “; volume, chapter, pages”. To handle empty
entries gracefully we cat them together and write the entire string afterwards.
1227 <JAmChemSoc | InorgChem> format.volume.noseries
1228 <ChemCommun | ChemEurJ | cv> ""
1229 }
1230 {
1231   format.crossref output.blank

Push an empty string since there will not be a volume.
1232   ""
1233 }
1234 if$
1235 format.chapter cat.comma
1236 format.pages cat.comma

```

```

1237   output
1238   format.note output
1239   fin.entry
1240 }

incollection
1241 FUNCTION {incollection}
1242 {
1243   output.bibitem
1244   format.authors "author" output.check
1245   ⟨cv⟩ format.title "title" output.check
1246   crossref missing$
1247   {
1248     format.in.booktitle "booktitle" output.check.blank
1249     format.edition
1250   ⟨ChemCommun | ChemEurJ | cv⟩      output.blank
1251   ⟨JAmChemSoc | InorgChem⟩        output.comma
1252   ⟨JAmChemSoc | InorgChem⟩        format.book.editors output.semicolon
1253     format.book.volume.and.series output
1254   ⟨ChemCommun | ChemEurJ | cv⟩      format.book.volume.noseries output
1255   ⟨ChemCommun | ChemEurJ | cv⟩      format.book.editors output.comma
1256     format.publisher.address output
1257     format.date "year" output.check.comma

```

The *J. Am. Chem. Soc.* requires “; volume, chapter, pages”. To handle empty entries gracefully we cat them together and write the entire string afterwards.

```

1258   ⟨JAmChemSoc | InorgChem⟩      format.volume.noseries
1259   ⟨ChemCommun | ChemEurJ | cv⟩      ""
1260   }
1261   {
1262     format.crossref output.blank

```

Push an empty string since there will not be a volume.

```

1263   ""
1264   }
1265   if$"
1266   format.chapter cat.comma
1267   format.pages cat.comma
1268   output
1269   format.note output
1270   fin.entry
1271 }

```

#### inproceedings

```

1272 FUNCTION {inproceedings}
1273 {
1274   output.bibitem
1275   format.authors "author" output.check
1276   ⟨cv⟩ format.title "title" output.check
1277   crossref missing$
1278   {
1279     format.in.booktitle "booktitle" output.check.blank
1280     publisher empty$
1281       { format.organization.address output }
1282       {

```

```

1283      organization bibinfo.check output
1284      format.publisher.address output
1285    }
1286    if$
1287    format.book.volume.and.series output
1288  }
1289  { format.crossref output.blank }
1290 if$
1291 format.pages "pages" output.check
1292 format.note output
1293 format.url output
1294 fin.entry
1295 }

manual
1296 FUNCTION {manual}
1297 {
1298   output.bibitem
1299   author empty$ 
1300   {
1301     organization bibinfo.check
1302     duplicate$ empty$ 
1303     'pop$ 
1304     {
1305       output
1306       address bibinfo.check output
1307     }
1308     if$
1309   }
1310   { format.authors output }
1311 if$
1312 format.title "title" output.check
1313 author empty$ 
1314 {
1315   organization empty$ 
1316   { address bibinfo.check output }
1317   'skip$ 
1318   if$
1319 }
1320 {
1321   organization bibinfo.check output
1322   address bibinfo.check output
1323 }
1324 if$
1325 format.edition output
1326 format.date output
1327 format.note output
1328 format.url output
1329 fin.entry
1330 }

mastersthesis
1331 FUNCTION {mastersthesis}
1332 {

```

```

1333   output.bibitem
1334   format.authors "author" output.check
1335   bbl.mthesis format.thesis.type output
1336   school "school" bibinfo.warn output
1337   address bibinfo.check output
1338   format.date "year" output.check
1339   format.note output
1340   format.url output
1341   fin.entry
1342 }

misc
1343 FUNCTION {misc}
1344 {
1345   output.bibitem
1346   format.authors output
1347   format.title output
1348   howpublished bibinfo.check output
1349   format.date output
1350   format.note output
1351   format.url output
1352   fin.entry
1353   misc.empty.check
1354 }

phdthesis
1355 FUNCTION {phdthesis}
1356 {
1357   output.bibitem
1358   format.authors "author" output.check
1359   <cv> format.title "title" output.check
1360   bbl.phdthesis format.thesis.type output
1361   school "school" bibinfo.warn output
1362   address bibinfo.check output
1363   format.date "year" output.check
1364   format.note output
1365   format.url output
1366   fin.entry
1367 }

proceedings
1368 FUNCTION {proceedings}
1369 {
1370   output.bibitem
1371   editor empty$ 
1372   { organization bibinfo.check output }
1373   { format.editors output }
1374   if$ 
1375   format.title "title" output.check
1376   format.volume output
1377   format.number.series output
1378   editor empty$ 
1379   { 
1380     publisher empty$ 

```

```

1381      'skip$  

1382      { format.publisher.address output }  

1383      if$  

1384    }  

1385    {  

1386      publisher empty$  

1387      { format.organization.address output }  

1388      {  

1389        organization bibinfo.check output  

1390        format.publisher.address output  

1391      }  

1392      if$  

1393    }  

1394    if$  

1395 <ChemCommun | ChemEurJ | cv> format.date "year" output.check  

1396    format.note output  

1397    format.url output  

1398    fin.entry  

1399 }  

  

program  

1400 FUNCTION {program}  

1401 {  

1402   output.bibitem  

1403   format.authors output  

1404   format.program.title "title"  

1405 <ChemCommun | ChemEurJ | cv> output.check  

1406 <JAChemSoc | InorgChem> output.check.blank  

1407   format.program.description output.comma  

1408   format.program.version output  

1409 <ChemCommun | ChemEurJ | cv> format.date output  

1410   format.program.publisher.address output  

1411 <JAChemSoc | InorgChem> format.date output.comma  

1412   format.note output  

1413   format.url output  

1414   fin.entry  

1415 }  

  

techreport  

1416 FUNCTION {techreport}  

1417 {  

1418   output.bibitem  

1419   format.authors "author" output.check  

1420   format.title  

1421   "title" output.check  

1422   format.tr.number output  

1423   institution "institution" bibinfo.warn output  

1424   address bibinfo.check output  

1425   format.date "year" output.check  

1426   format.note output  

1427   format.url output  

1428   fin.entry  

1429 }

```

```

unpublished
1430 FUNCTION {unpublished}
1431 {
1432   output.bibitem
1433   format.authors "author" output.check
1434   format.title "title" output
1435   format.date output
1436   format.note "note" output.check
1437   format.url output
1438   fin.entry
1439 }

conference conference is an alias for inproceedings.
1440 FUNCTION {conference} { inproceedings }

default.type The default type is misc.
1441 FUNCTION {default.type} { misc }

```

## 4.12 Main program

**begin.bib** This function starts the bibliography by opening the `thebibliography` environment.

```

1442 FUNCTION {begin.bib}
1443 {
1444   "\begin{thebibliography}{\quad * {\quad} * \newline$"
1445 }

```

**end.bib** This function ends the bibliography by closing the `thebibliography` environment.

```

1446 FUNCTION {end.bib}
1447 {
1448   \newline$"
1449   "\end{thebibliography}" \newline$"
1450 }

```

**initialize.longest.label** This function sets up the variables for the longest label.

```

1451 FUNCTION {initialize.longest.label}
1452 {
1453   "" 'longest.label :=
1454   #1 'number.label :=
1455   #0 'longest.label.width :=
1456 }

```

**init.consts** This function sets up some constant used while formatting the bibliography.

```

1457 FUNCTION {init.consts}
1458 {
1459   #0 'before.all :=
1460   #1 'mid.sentence :=
1461   #2 'new.sentence :=
1462 }

```

**longest.label.pass** This function is used to determine the length of the longest label. It checks whether the current label is longer than the current longest label. If so, it updates `longest.label.width`.

```

1463 FUNCTION {longest.label.pass}
1464 {
1465   number.label int.to.str$ 'label :=
1466   number.label #1 + 'number.label :=
1467   label width$ longest.label.width >
1468   {
1469     label 'longest.label :=
1470     label width$ 'longest.label.width :=
1471   }
1472   'skip$
1473   if$
1474 }

```

`write.babel.misc` This function writes the default definitions of some commands used in formatting a citation.

```

1475 FUNCTION {write.babel.misc}
1476 {
1477   "\providecommand{\bbland}{and}" write$ newline$
1478   "\providecommand{\bbchap}{chap.}" write$ newline$
1479   "\providecommand{\bbchapter}{chapter}" write$ newline$
1480   "\providecommand{\bbletal}{et~al.}" write$ newline$
1481   "\providecommand{\bbleditors}{editors}" write$ newline$
1482   "\providecommand{\bbleds}{eds.}" write$ newline$
1483   "\providecommand{\bbleditor}{editor}" write$ newline$
1484   "\providecommand{\bbled}{ed.}" write$ newline$
1485   "\providecommand{\bbledition}{edition}" write$ newline$
1486   "\providecommand{\bbledn}{ed.}" write$ newline$
1487   "\providecommand{\bbleidp}{page}" write$ newline$
1488   "\providecommand{\bbleidpp}{pages}" write$ newline$
1489   "\providecommand{\bbleratum}{erratum}" write$ newline$
1490   "\providecommand{\bblin}{in}" write$ newline$
1491   "\providecommand{\bblmthesis}{Master's thesis}" write$ newline$
1492   "\providecommand{\bblno}{no.}" write$ newline$
1493   "\providecommand{\bblnumber}{number}" write$ newline$
1494   "\providecommand{\bblof}{of}" write$ newline$
1495   "\providecommand{\bblpage}{page}" write$ newline$
1496   "\providecommand{\bblpages}{pages}" write$ newline$
1497   "\providecommand{\bblp}{p}" write$ newline$
1498   "\providecommand{\bbldthesis}{Ph.D. thesis}" write$ newline$
1499   "\providecommand{\bblp}{pp}" write$ newline$
1500   "\providecommand{\bbltechrep}{Tech. Rep.}" write$ newline$
1501   "\providecommand{\bbltechreport}{Technical Report}" write$ newline$
1502   "\providecommand{\bblvolume}{volume}" write$ newline$
1503   "\providecommand{\bblvol}{Vol.}" write$ newline$
1504 }

```

`write.babel.months` This function writes the default names of the months.

```

1505 FUNCTION {write.babel.months}
1506 {
1507   "\providecommand{\bbjan}{January}" write$ newline$
1508   "\providecommand{\bbfeb}{February}" write$ newline$
1509   "\providecommand{\bbmar}{March}" write$ newline$
1510   "\providecommand{\bbapr}{April}" write$ newline$
1511   "\providecommand{\bbmay}{May}" write$ newline$

```

```

1512   "\providecommand{\bbbjun}{June}" write$ newline$
1513   "\providecommand{\bbbjul}{July}" write$ newline$
1514   "\providecommand{\bbblaug}{August}" write$ newline$
1515   "\providecommand{\bblsept}{September}" write$ newline$
1516   "\providecommand{\bbloct}{October}" write$ newline$
1517   "\providecommand{\bblnov}{November}" write$ newline$
1518   "\providecommand{\bbldec}{December}" write$ newline$
1519 }

```

`write.babel.ordinal`s This function writes the default names of the ordinals.

```

1520 FUNCTION {write.babel.ordinal}
1521 {
1522   "\providecommand{\bbbfir}{First}" write$ newline$
1523   "\providecommand{\bbbfirsto}{1st}" write$ newline$
1524   "\providecommand{\bbblsec}{Second}" write$ newline$
1525   "\providecommand{\bbblsecondo}{2nd}" write$ newline$
1526   "\providecommand{\bbblthir}{Third}" write$ newline$
1527   "\providecommand{\bbblthirdo}{3rd}" write$ newline$
1528   "\providecommand{\bbbfour}{Fourth}" write$ newline$
1529   "\providecommand{\bbbfourtho}{4th}" write$ newline$
1530   "\providecommand{\bbbfifth}{Fifth}" write$ newline$
1531   "\providecommand{\bbbfiftho}{5th}" write$ newline$
1532   "\providecommand{\bbblst}{st}" write$ newline$
1533   "\providecommand{\bbblnd}{nd}" write$ newline$
1534   "\providecommand{\bbblrnd}{rd}" write$ newline$
1535   "\providecommand{\bbblth}{th}" write$ newline$
1536 }

```

`write.babel` This function writes the default of all words used for formatting a citation.

```

1537 FUNCTION {write.babel}
1538 {
1539   write.babel.misc
1540   write.babel.months
1541   write.babel.ordinal
1542 }

```

`write.commands` This function writes the commands used in a citation.

```

1543 FUNCTION {write.commands}
1544 {
1545   "\providecommand{\url}[1]{\texttt{\#1}}" write$ newline$
1546   "\providecommand{\urlprefix}{}" write$ newline$
1547   "\providecommand{\foreignlanguage}[2]{\#2}" write$ newline$
1548   "\providecommand{\Capitalize}[1]{\uppercase{\#1}}" write$ newline$
1549   "\providecommand{\capitalise}[1]{\expandafter\Capitalize{\#1}}" write$ newline$
1550   "\providecommand{\bibliographycite}[1]{\cite{\#1}}" write$ newline$
1551   write.babel
1552 }

```

`write.header` This function writes the header of the .bbl file. It does not, however, start the `\thebibliography` environment. The contents of `@preamble` is always written first.

```

1553 FUNCTION {write.header}
1554 {
1555   preamble$ empty$
1556   'skip$'

```

```

1557     { preamble$ write$ newline$ }
1558     if$
1559     write.commands
1560 }

main Read all required entries from the database. The entries are in citation order.
1561 READ
    Determine the length of the longest label.
1562 EXECUTE {initialize.longest.label}
1563 ITERATE {longest.label.pass}

    Write the header to the file and open the thebibliography environment.
1564 EXECUTE {init.consts}
1565 EXECUTE {write.header}
1566 EXECUTE {begin.bib}

    Format all entries
1567 ITERATE {call.type$}

    Close the thebibliography environment.
1568 EXECUTE {end.bib}

```

## Change History

0.1.0		Renamed european bst to ChemEurJ bst . . . . .	1
General: Initial revision . . . . .	1		
0.1.1		<b>format.article.cat.journal.year:</b> NEW . . . . .	27
<b>format.chapter:</b> added capitalize to bbl.chapter for JACS . . . . .	21		
0.1.2		<b>format.article.erratum:</b> Use for- mat.article.cat.journal.year . . . . .	30
General: Added support for a cur- riculum vitae . . . . .	1	<b>format.article.erratum.germanpages:</b>	
Removed unused function add.blank . . . . .	1	Use format.article.cat.journal.year . . . . .	29
<b>init.consts:</b> Added new.sentence initialization . . . . .	41	<b>format.article.germanpages:</b> Use format.article.cat.journal.year . . . . .	31
<b>INTEGERS:</b> Removed unused vari- ables, added new.sentence . . . . .	8		
<b>format.article.title:</b> NEW . . . . .	32		
<b>output.internal:</b> Removed dead code (variable never got specific value) . . . . .	9		
0.1.3		<b>phdthesis:</b> Added title for style cv . . . . .	39
<b>select.language:</b> leave empty string on stack (replace pop\$ by skip\$) . . . . .	18	<b>incollection:</b> Added title for style cv . . . . .	37
0.2.0		<b>inproceedings:</b> Added title for style cv . . . . .	37
General: Added support for Chem. Commun. . . . .	1	Fixed formating of volume and series . . . . .	37
Renamed american bst to JAm- ChemSoc bst . . . . .	1	<b>write.commands:</b> Replaced new- command by providecommand for capitalize . . . . .	43
0.2.2			
		General: Added InorgChem bst . . . . .	1
		JAmChemSoc now requires truncation of author list after more than 15 names only. . . . .	1

<b>format.names:</b> Added fifteennames option . . . . .	15	iter after last author name . . . . .	40
<b>0.2.3</b>		<b>0.2.4</b>	
General: Fixed some minor issues with ACS styles. . . . .	1	General: Removed number of pages from papers with EID in all styles except 'cv'. . . . .	1
<b>format.editors:</b> ACS does not print 'Eds.' if there are no authors. . . . .	22	<b>format.article.eid:</b> Include number of pages only for style 'cv'	28
<b>format.url:</b> Put URL into a new sentence. . . . .	25	<b>0.2.5</b>	
<b>program:</b> ACS does not like a delim-		General: Minor documentation updates . . . . .	1

## Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

	<b>A</b>		extract.num . . . . .	292
and . . . . .		178		
article . . . . .		<u>1154</u>		
	<b>B</b>			
bbl.* . . . . .		71		
begin.bib . . . . .		<u>1442</u>		
bibinfo.check . . . . .		207		
bibinfo.warn . . . . .		214		
bibliography.cite . . . . .		829		
bolden . . . . .		233		
book . . . . .		<u>1178</u>		
booklet . . . . .		<u>1198</u>		
	<b>C</b>			
capitalize . . . . .		240		
cat.blank . . . . .		260		
cat.colon . . . . .		260		
cat.comma . . . . .		260		
cat.default . . . . .		260		
cat.internal . . . . .		244		
cat.semicolon . . . . .		260		
conference . . . . .		1440		
convert.edition . . . . .		311		
	<b>D</b>			
default.type . . . . .		<u>1441</u>		
delimiter.* . . . . .		112		
	<b>E</b>			
either.or.check . . . . .		340		
emphasize . . . . .		347		
end.bib . . . . .		<u>1446</u>		
eng.ord . . . . .		265		
ENTRY . . . . .		8		
	<b>F</b>			
fin.entry . . . . .		354		
format.article.cat.journal.year . . . . .		839		
format.article.date . . . . .		878		
format.article.eid . . . . .		896		
format.article.erratum . . . . .		966		
format.article.erratum.eid . . . . .		956		
format.article.erratum.germanpages . . . . .		938		
format.article.erratum.germanpages.pages . . . . .		922		
format.article.erratum.germanpages.volume . . . . .		933		
format.article.erratum.journal . . . . .		950		
format.article.erratum.pages . . . . .		951		
format.article.erratum.volume . . . . .		961		
format.article.erratum.year . . . . .		927		
format.article.germanpages . . . . .		998		
format.article.germanpages.journalname . . . . .		917		
format.article.germanpages.pages . . . . .		987		
format.article.germanpages.pages.internal . . . . .		874		
format.article.germanpages.volume . . . . .		992		
format.article.germanpages.volume.internal . . . . .		853		
format.article.germanpages.year . . . . .		997		
format.article.journal . . . . .		909		
format.article.numpages . . . . .		883		
format.article.pages . . . . .		1010		
format.article.title . . . . .		1028		

format.article.volume.and.number		M	
.....	1038	main .....	1561
format.article.volume.internal	848	manual .....	1296
format.article.year.internal	844	mastersthesis .....	1331
format.authors	588	misc .....	1343
format.book.authors	1063	misc.empty.check .....	1115
format.book.editors	1073	multi.page.check .....	432
format.book.volume.and.series	1091	multiply .....	190
format.book.volume.internal	1080		
format.book.volume.noseries	1105	N	
format.booktitle	603	n.dashify .....	460
format.chapter	609	not .....	172
format.crossref	833		
format.date	627	O	
format.edition	632	or .....	184
format.editors	648	output .....	164
format.in.booktitle	668	output.bibitem .....	121
format.names	359	output.blank .....	164
format.note	680	output.check .....	168
format.number.series	684	output.check.blank .....	168
format.org.or.pub	710	output.check.comma .....	168
format.organization.address	728	output.check.internal .....	155
format.pages	733	output.check.semicolon .....	168
format.program.description	1124	output.comma .....	164
format.program.publisher.address	1132	output.internal .....	131
format.program.title	1137	output.semicolon .....	164
format.program.version	1150		
format.publisher.address	747	P	
format.thesis.type	752	phdthesis .....	1355
format.title	763	proceedings .....	1368
format.tr.number	774	program .....	1400
format.url	789		
format.volume	800	S	
format.volume.and.series	805	select.language .....	491
format.volume.internal	576	space.word .....	503
format.volume.noseries	818	str.to.int .....	511
format.year.internal	572	str.to.int.warn .....	507
		STRINGS .....	64
G		T	
get.bbl.editor	420	techreport .....	1416
get.bbl.erratum	427	tie.or.space.prefix .....	560
get.bbl.page	452		
I		U	
inbook	1209	unpublished .....	1430
incollection	1241		
init.consts	1457	W	
initialize.longest.label	1451	word.in .....	568
inproceedings	1272	writebabel .....	1537
INTEGERS	49	writebabel.misc .....	1475
is.num	286	writebabel.months .....	1505
L		writebabel.ordinal .....	1520
longest.label.pass	1463	write.commands .....	1543
		write.header .....	1553