

# The GOST package

Igor A. Kotelnikov\*

2012/02/13, v.1.2

## Abstract

GOST is a bundle of BibTeX styles designed to meet State Standards (GOST) on information, librarianship and publishing issued by The Russian Federation and Interstate Committee of former USSR States.

It comprises 16 BibTeX styles to format bibliography in English, Russian and Ukrainian according to GOST 7.0.5-2008 and GOST 7.1-2003. Both 8-bit and Unicode (UTF-8) versions of each BibTeX style, in each case offering a choice of sorted and unsorted. Further, 2 more styles, `gost780` and `gost780s` styles (not conforming to effective standards) are retained for backwards compatibility.

## 1 Introduction

The package was initially developed by Maksym Polyakov. It was later updated by Igor Kotelnikov to the present status and some code was borrowed from `disser` package developed by Stanislav Kruchinin and unpublished work by Artem Petrenkov.

Nowdays, GOST is a bundle of BibTeX styles designed to meet State Standards (GOST) on information, librarianship and publishing issued by Russian Federation and interstate committee of former USSR States.

The System of Standards includes:

**GOST 7.0.5-2008** Bibliographic reference. General requirements and rules of making.

**GOST 7.1 -2003** Bibliographic record. Bibliographic description. General requirements and rules.

**GOST 7.80 -2000** Bibliographic record. Heading. General requirements and rules.

**GOST 7.11 -2004** Bibliographic description and references. Rules for the abbreviation of words and word combinations in foreign European languages.

**GOST 7.83 -2001** Electronic editions. Basic types and imprint.

ect.

Currently, GOST contains 16 BibTeX styles to format bibliography in English, Russian and Ukrainian according to GOST 7.0.5-2008 and GOST 7.1-2003. Both 8-bit and Unicode (UTF-8)

---

\*kia999@mail.ru

versions of each BibTeX style, in each case offering a choice of sorted and unsorted. Further, 2 styles, `gost780` and `gost780s` styles (which do not conform to effective standards) are retained for backwards compatibility.

All styles in the GOST bundle are derived from single master file `gost.dtx` by applying different set of options as shown in the table below.

Style	utf8	strict	modern	eprint	long	sort	natbib
<code>gost780</code>							
<code>gost780s</code>						+	
<code>gost2003</code>		+		+			
<code>gost2003s</code>		+		+		+	
<code>gost2008</code>			+	+			
<code>gost2008n</code>			+	+			+
<code>gost2008l</code>			+	+	+		
<code>gost2008s</code>			+	+		+	
<code>gost2008ns</code>			+	+		+	+
<code>gost2008ls</code>			+	+	+	+	
<code>ugost2003</code>	+	+		+			
<code>ugost2003s</code>	+	+		+		+	
<code>ugost2008</code>	+		+	+			
<code>ugost2008n</code>	+		+	+			+
<code>ugost2008l</code>	+		+	+	+		
<code>ugost2008s</code>	+		+	+		+	
<code>ugost2008ns</code>	+		+	+		+	+
<code>ugost2008ls</code>	+		+	+	+	+	
Style	utf8	strict	modern	eprint	long	sort	natbib

`Gost2008` style is recommended for most applications. It corresponds to the currently effective Standard 7.0.5-2008. Librarians should use the style `gost2003` instead of `gost2008` to compile a library catalog to meet the Standard 7.1-2003. Use of other styles is best explained through the meaning of options used to compile those styles from the master source.

Two styles, `gost780` and `gost780s`, compiled without `modern` and `strict` options, are retained for backward compatibility. They do not conform the Standards 7.0.5-2008 and 7.1-2003 cited above.

The `strict` option provides conformance to the Standard 7.1-2003. The styles compiled with that option bear the name `gost2003` with possible suffixes `s`, `l`, `n` as explained below. These styles are intended primarily for the librarians who compose a library catalog.

The `modern` option meets the Standard 7.0.5-2008 which can be thought off as a relaxed version of the Standards 7.1-2003. The styles compiled with that option bear the name `gost2008` with possible suffixes `s`, `l`, `n`. The `strict` option has precedence over `modern` so that a style compiled with both options will resemble mainly the `gost2003` style rather than `gost2008`.

If the number of authors exceeds 4, modern styles cut the list of authors to at most 4 persons as prescribed by the Standards. The option `long` overrides this rule to provide backward compatibility with the package `disser` by Stanislav Kruchinin. Two styles, `gost2008l` and `gost2008ls`, compiled with the option `long` mimic behavior of the styles `gost705` and `gost705s` from the `disser` package. Major effect of the `long` option is that the list of authors always precedes book or article title no matter how long is it. Modern styles compiled without `long` place long list of authors behind the title. The names of styles compiled with the option `long` has the suffix `l`. Recall that those styles do not conform effective Standards and their use is discouraged.

The `eprint` option enables formatting electronic publications. In particular, it enables `eprint`, `eprinttype`, `eprintclass`, and `doi` fields for a bibliographic entry. The styles generated without the `eprint` option, ignore these fields. Starting from the version 1.2 of the GOST package, all modern styles are compiled with this option included, and the suffix `e` which designated this option in earlier versions is not appended to the name of style any more.

The `natbib` option provides compatibility with the `natbib` package. The names of styles compiled with the option `natbib` bear the suffix `n`. Currently 4 styles with that option are available for beta testing.

The `sort` option enables sorting bibliographic references by author names and references titles. The names of styles compiled with the option `sort` bear the suffix `s`.

Finally, the `utf8` option produces bibliographic styles in unicode rather than in 8-bit encoding. Names of those styles bear the prefix `u`.

Beyond bibliographic style, GOST bundle contains CS files (codepage and sorting order).

Encoding		CSF	Sorting order
cp866		ruscii.csf	Cyrillic first, Latin
cp1251		cp1251.csf	Cyrillic first, Latin
koi8-u		koi8u.csf	Cyrillic first, Latin
utf8		utf8cyrillic.csf	Cyrillic first, Latin

In addition, BibTeX8 distribution comes with few more CSFs.

Encoding		CSF	Sorting order
cp866		cp866rus.csf	Latin first, Cyrillic

## 1.1 How to use

1. Select bibliography style by adding appropriate `\bibliographystyle` declaration to your source file `<filename>.tex`, e.g.

```
\bibliographystyle{gost2008}
\bibliography{database}
```

2. Add the field `language="ukrainian"` or `language="russian"` to the bibliographic entries in Ukrainian or Russian languages in your database; English is the default language. German, Italian and French are partially supported.
3. To compile list of references from your database use `bibtex8.exe` rather than `bibtex.exe`. Depending on the codepage of your bibliographic database, indicate one of the CS files listed above as option to `bibtex8.exe`. Run LaTeX, then run `bibTeX8` and LaTeX again:

```
latex <filename>.tex
bibtex8 -B -c <csf_file>.csf <filename>.aux
latex <filename>.tex
```

4. For details on preparing bibliographic database see examples in `gost*.pdf`.
5. `ugost*` styles are primarily intended for use with unicode compilers (`xelatex` and `lualatex`). They should be preferred as well when using 8bit compilers (`latex` and `pdflatex`) if source file is in utf8 encoding.
6. Neither `bitex.exe` nor `bitex8.exe` provides correct sorting order of unicode text. It means that using `ugost2008s` or `ugost2008ns` may produce unexpected result for documents in utf8 encoding.
7. `Bibtex8` fails to change case of a string if it contains Cyrillic letter in unicode. Therefore `ugost2008*` styles do not change case of titles and other parts of bibliographic record while 8-bit styles do the case change where appropriate.
8. Either `bibtex8` or `Bibtex8` fail to cut Cyrillic names to initials. Therefore `ugost2008*` styles do not modify name of authors.
9. Package `natbib` is required when choosing styles with suffix `n` in their names.

## 1.2 Customization

Every GOST style defines few commands to format some parts of a reference. You can redefine these commands prior to the `\bibliography{<bibtex_style>}` command. Initial definitions are listed below.

```
\providecommand*{\url}[1]{\small #1}
\providecommand*{\BibUrl}[1]{\url{#1}}
\providecommand*{\BibAnnote}[1]{}
\providecommand*{\BibEmph}[1]{#1}
```

By default, `gost` styles separate logical parts of a bibliography record by a period and cyrdash (`. ---`). It is legitimate to drop that dash by overriding the command `\BibDash` as follows

```
\providecommand*{\BibDash}{}%
```

By default, `\BibDash` is equivalent to the shorthand `---` defined by the `babel` package with the option `ruussian`. It prints a so called Cyrillic dash (`\cyrdash`), which is 20 shorter than ordinary LaTeX dash (`---`), and puts unbreakable space before `\cyrdash` so that dash never appears in the beginning of a line.

### 1.3 Where to get

1. [CTAN:biblio/bibtex/contrib/gost](#).
2. [CTAN:pkg/gost](#).

### 1.4 What's new in version 1.2 (2012.02.22)

1. Code refactoring. All styles are now generated from single source file.
2. Support for GOST-7.1-2003. The field `medium` is added to reflect type of material. For most entry types `medium` defaults to `text`.
3. Support for `natbib` package.
4. Support for `natbib` package.
5. All modern styles are now compiled with the `eprint` option.

### 1.5 What's new in version 1.1 (2012.01.21)

1. Support for GOST 7.0.5-2008 and GOST 7.1-2003 is provided.
2. `@Online` entry is added to format a reference to electronic resource on Internet.
3. `@MastersThesis` entry is added to format a reference to master's thesis.
4. `@DSciThesis` entry is added to format a reference to doctor of sciences thesis.
5. `Urldate`, `eprint`, `eprintclass`, `eprinttype` fields are added.

### 1.6 Version history

2012.02.22 Support for `natbib` package.

2012.02.02 Adaptation to GOST 7.0.5, electronic publishing.

2005.08.12 First version uploaded to CTAN.

2003.06.06 First public version.

## 2 Implementation

We need Russian fonts to produce documentation of the code below. Therefore we switch current language to Russian by issuing the command `\selectlanguage{russian}`.

```
1 (*bst)
2 %%
3 %% This bibstyle attempts to format bibliography according to
4 (!modern)%% GOST 7.80-2000 for bibliographic records.
5 (modern)%% GOST 7.0.5-2008 for bibliographic reference.
6 (*natbib)%%
7 %%-----
8 %% This is an author-year citation style bibliography.
9 %% It requires a special package file to function properly
10 %% such as natbib.sty by Patrick W. Daly.
11 %% The form of the \bibitem entries is
12 %% \bibitem[Jones et al.(1990)]{key}...
13 %% \bibitem[Jones et al.(1990)Jones, Baker, and Smith]{key}...
14 %% where the label part (in brackets) consists of the author names,
15 %% as they should appear in the citation, with the year in parentheses following.
16 %% There must be no space before the opening parenthesis!
17 %% A full list of authors may also follow the year.
18 %% In natbib.sty, it is possible to define the type of enclosures that is
19 %% really wanted (brackets or parentheses), but in either case, there must
20 %% be parentheses in the label.
21 %% The \cite command functions as follows:
22 %% \citet{key} => Jones et al. (1990)
23 %% \citet*{key} => Jones, Baker, and Smith (1990)
24 %% \citep{key} => (Jones et al., 1990)
25 %% \citep*{key} => (Jones, Baker, and Smith, 1990)
26 %% \citep[chap. 2]{key} => (Jones et al., 1990, chap. 2)
27 %% \citep[e.g.][] {key} => (e.g. Jones et al., 1990)
28 %% \citep[e.g.][p. 32]{key} => (e.g. Jones et al., p. 32)
29 %% \citeauthor{key} => Jones et al.
30 %% \citeauthor*{key} => Jones, Baker, and Smith
31 %% \citeyear{key} => 1990
32 %%-----
33 (/natbib)
34
```

### 2.1 Fields

Enlist all entry types allowed in a bibliographic database. Most entries are common for many standard bst styles.

```
35 ENTRY
36 { address
37   annote
38   author
39   booktitle
40   bookauthor
```

```

41 chapter
42 edition
43 editor
44 compiler
45 howpublished
46 institution
47 journal
48 key
49 month
50 note
51 number
52 organization
53 pages
54 publisher
55 school
56 series
57 title
58 medium % new in v.2.
59 type
60 volume
61 year
62 language
63 booklanguage

```

Remaining entries are borrowed from biblatex.

```

64 numpages
65 url
66 urldate
67 isbn
68 doi
69 % archive
70 eprinttype % = archivePrefix
71 eprintclass % = primaryClass
72 eprint
73 }
74 {}
75 <natbib> { label }
76 <natbib> { label extra.label sort.label short.list }
77

```

Declare internal variables and constants used in to format references.

```

78 INTEGERS { output.state before.all mid.sentence after.sentence after.block
79 after.dblslash after.slash after.column after.semicolumn }
80

```

init.state.consts

```

81 FUNCTION {init.state.consts}
82 { #0 'before.all :=
83   #1 'mid.sentence :=
84   #2 'after.sentence :=
85   #3 'after.block :=
86   #4 'after.dblslash :=

```

```

87 #5 'after.slash :=
88 #6 'after.column :=
89 #7 'after.semicolumn :=
90 }
91
92 STRINGS { s t }
93
94 STRINGS { curlanguage }
95

```

## 2.2 Formatting functions

`change.language`     Declare function to switch language.

```

96 FUNCTION {change.language}
97 { booklanguage empty$
98   { "" }
99   { booklanguage 'curlanguage :=
100     "\selectlanguageifdefined{"
101     curlanguage *
102     "}" *
103   }
104   if$
105 }
106

```

`output.nonnull`     Declare functions to output various parts of bibliographic record.

```

107 FUNCTION {output.nonnull}
108 {
109   swap$
110   output.state mid.sentence =
111   { ", " * write$ }
112   { output.state after.block =
113     { add.period$ write$
114     (!modern)      " " quote$ "--- " * * write$
115     (modern)      " \BibDash " write$
116     newline$
117     "\newblock " write$
118   }
119   { output.state before.all =
120     'write$
121     { output.state after.dblslash =
122       { "~/ " * change.language * " " * write$ }
123     { output.state after.slash =
124       { "~/ " * write$ }
125     { output.state after.column =
126     (!strict | modern)) { ": " * write$ }
127     (strict | modern)  { "~: " * write$ }
128     { output.state after.semicolumn =
129     (!strict | modern)) { "; " * write$ }
130     (strict | modern)  { "~; " * write$ }

```



```

131             { add.period$ " " * write$ }
132             if$
133             }
134         if$
135     }
136     if$
137     }
138     if$
139     }
140     if$
141     }
142     if$
143     mid.sentence 'output.state :=
144     }
145     if$
146 }
147
output
output.check 148 FUNCTION {output}
149 { duplicate$ empty$
150     'pop$
151     'output.nonnull
152     if$
153 }
154
155 FUNCTION {output.check}
156 { 't :=
157     duplicate$ empty$
158     { pop$ "empty " t * " in " * cite$ * warning$ }
159     'output.nonnull
160     if$
161 }
162
fin.entry    fin.entry finalizes current entry. It writes dot, if no dot is found in stack, and starts new line.
163 FUNCTION {fin.entry}
164 { add.period$
165     write$
166     newline$
167 }
168
new.block    Declare family of functions to put punctuation marks depending of current status of output
stack. The just check output state and revert it another state if required. Checking output state
prevents occasional doubling of punctuation marks.
169 FUNCTION {new.block}
170 { output.state before.all =
171     'skip$
172     { after.block 'output.state := }
173     if$

```

```
174 }
175
```

new.dblslash

```
176 FUNCTION {new.dblslash}
177 { output.state before.all =
178   'skip$
179   { after.dblslash 'output.state := }
180   if$
181 }
182
```

new.slash

```
183 FUNCTION {new.slash}
184 { output.state before.all =
185   'skip$
186   { after.slash 'output.state := }
187   if$
188 }
189
```

new.column

```
190 FUNCTION {new.column}
191 { output.state before.all =
192   'skip$
193   { after.column 'output.state := }
194   if$
195 }
196
```

new.semicolumn

```
197 FUNCTION {new.semicolumn}
198 { output.state before.all =
199   'skip$
200   { after.semicolumn 'output.state := }
201   if$
202 }
203
```

new.sentence

```
204 FUNCTION {new.sentence}
205 { output.state after.block =
206   'skip$
207   { output.state before.all =
208     'skip$
209     { after.sentence 'output.state := }
210     if$
211   }
212   if$
213 }
214
```

```

add.blank
215 FUNCTION {add.blank}
216 { " " * before.all 'output.state :=
217 }
218

not      Declare few logical functions.
219 FUNCTION {not}
220 { { #0 }
221   { #1 }
222   if$
223 }
224

and
225 FUNCTION {and}
226 { 'skip$
227   { pop$ #0 }
228   if$
229 }
230

or
231 FUNCTION {or}
232 { { pop$ #1 }
233   'skip$
234   if$
235 }
236

chop.word  The function chop.word in substr len str chop.word removes given substring substr of length
           len from the beginning of the string str.
237 <sort | natbib>
238 INTEGERS { len }
239
240 FUNCTION {chop.word}
241 { 's :=
242   'len :=
243   s #1 len substring$ =
244     { s len #1 + global.max$ substring$ }
245   's
246   if$
247 }
248 </sort | natbib>
249

non.stop
250 FUNCTION {non.stop}
251 { duplicate$
252   "]" * add.period$
253   #-1 #1 substring$ "." =
254 }

```

```

255
new.block.checka
256 FUNCTION {new.block.checka}
257 { empty$
258   'skip$
259   'new.block
260   if$
261 }
262
new.block.checkb
263 FUNCTION {new.block.checkb}
264 { empty$
265   swap$ empty$
266   and
267   'skip$
268   'new.block
269   if$
270 }
271
w.sentence.checka
272 FUNCTION {new.sentence.checka}
273 { empty$
274   'skip$
275   'new.sentence
276   if$
277 }
278
w.sentence.checkb
279 FUNCTION {new.sentence.checkb}
280 { empty$
281   swap$ empty$
282   and
283   'skip$
284   'new.sentence
285   if$
286 }
287
w.dblslash.checka   For online entry.
288 FUNCTION {new.dblslash.checka}
289 { empty$
290   'skip$
291   'new.dblslash
292   if$
293 }
294
field.or.null
295 FUNCTION {field.or.null}

```

```

296 { duplicate$ empty$
297   { pop$ "" }
298   'skip$
299   if$
300 }
301
emphasize      Declare function to emphasize last string in stack.
302 FUNCTION {emphasize}
303 { duplicate$ empty$
304   { pop$ "" }
305   { "\BibEmph{" swap$ * "}" * }
306   if$
307 }
308
e.square.brackets      New in v.1.1.2. Declare function to enclose last word in square brackets. It do so only for the
                        string option.
309 <!*strict>
310 FUNCTION {enclose.square.brackets} { }
311 </!strict>
312 <*strict>
313 FUNCTION {enclose.square.brackets}
314 { duplicate$ empty$
315   { pop$ "" }
316   { "[" swap$ * "]" * }
317   if$
318 }
319 </strict>
320
321
322 % \DescribeFunction{bracify}
323 % \DescribeFunction{bracketise}
324 % \DescribeFunction{parenthesify}
325 % New in v.1.1.2. An idea borrowed from apsrev4-1.bst.
326 % Declare function to enclose last word in braces, square brackets and
327 % parenthesis.
328 % \begin{macrocode}
329 FUNCTION {bracify}
330 { duplicate$ empty$
331   { pop$ "{}" }
332   { "{" swap$ * "}" * }
333   if$
334 }
335 FUNCTION {bracketise}
336 { duplicate$ empty$
337   { pop$ "[]" }
338   { "[" swap$ * "]" * }
339   if$
340 }
341 FUNCTION {parenthesify}

```

```

342 { duplicate$ empty$
343   { pop$ "(" }
344   { "(" swap$ * " " * }
345   if$
346 }
347
348

```

space.word      space.word inserts space before last string in stack.

```

349 FUNCTION {space.word}
350 { " " swap$ * " " * }
351

```

## 2.3 Standard abbreviations

bbl.edby    Declare language-sensitive abbreviations.

```

352 FUNCTION {bbl.edby}    % { "\bbledby{}" }
353 { curlanguage "english" =
354   {"Ed.\ by"}
355   { curlanguage "ukrainian" =
356     {!utf8}        {"{\cyr\CYRP\cyrii\cyrd\ \cyrr\cyre\cyrd.}" }
357     {utf8}        {"Під ред."}
358     { curlanguage "russian" =
359       {!utf8}        {"{\cyr\CYRP\cyro\cyrd\ \cyrr\cyre\cyrd.}" }
360       {utf8}        {"Под ред."}
361       { curlanguage "german" =
362         { "ed." }
363         {"language is not defined: " language "edby" * * warning$ "Ed.\ by"}
364         if$}
365       if$}
366     if$}
367 if$}
368

```

bbl.cmplr

```

369 FUNCTION {bbl.cmplr}
370 { curlanguage "english" =
371   { "Compiler" }
372   { curlanguage "german" =
373     { "Hrsg." }
374     { curlanguage "ukrainian" =
375       {!utf8}        {"{\cyr\CYRU\cyrk\cyrl.}" }
376       {utf8}        {"{Укл.}" }
377       { curlanguage "russian" =
378         {!utf8}        {"{\cyr\CYRS\cyro\cyrs\cyrt.}" }
379         {utf8}        {"{Сост.}" }
380         {"language is not defined: " language "cmplr" * * warning$ "Compiler"}
381         if$}
382       if$}
383     if$}
384 if$}

```

385

bb1.edition

```
386 FUNCTION {bb1.edition} % { "\bb1edition{}" }
387 { curlanguage "english" =
388   {"edition"}
389   { curlanguage "ukrainian" =
390     <utf8> {"{\cyr\cyrv\cyri\cyrd.}" }
391     <utf8> {"{\ВИД.}" }
392     { curlanguage "russian" =
393       <utf8> {"{\cyr\cyri\cyrz\cyrd.}" }
394       <utf8> {"{\изд.}" }
395       { curlanguage "german" =
396         {"{aus.}" } %%% { "Auf1." } ??
397         { curlanguage "italian" =
398           {"edizione"}
399           { curlanguage "french" =
400             {"\'{e}dition"}
401             {"language is not defined: " language "edition" * * warning$ "edition"}
402             if$}
403             if$}
404             if$}
405             if$}
406             if$}
407 if$}
408
```

bb1.vvolume

```
409 FUNCTION {bb1.vvolume} % { "\bb1Volume{}" }
410 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
411   {"Volume"}
412   { curlanguage "ukrainian" = curlanguage "russian" = or
413     <utf8> {"{\CYRT\cyro\cyrm" }
414     <utf8> {"{Tom" }
415     { curlanguage "german" =
416       {"{Band}" } %%% { "Volumen" }
417       {"language is not defined: " language "vvolume" * * warning$ "Volume"}
418       if$}
419       if$}
420 if$}
421
```

bb1.vvol

```
422 FUNCTION {bb1.vvol} % { "\bb1Vol{}" }
423 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
424   {"Vol."}
425   { curlanguage "ukrainian" = curlanguage "russian" = or
426     <utf8> {"{\CYRT."}
427     <utf8> {"{T."}
428     { curlanguage "german" =
429       {"{Bd.}" } %%% { "Vol." }
430       {"language is not defined: " language "vvol" * * warning$ "Vol."}

```

```

431     if$}
432   if$}
433 if$}
434

```

bbl.iissue

```

435 FUNCTION {bbl.iissue} % { "\bblIssue{" }
436 { curlanguage "english" =
437   {"Issue"}
438   { curlanguage "ukrainian" =
439     ⟨!utf8⟩ {"\CYRV\cyri\cyrp\cyru\cyrs\cyrk"}
440     ⟨utf8⟩ {"Випуск"}
441     { curlanguage "russian" =
442       ⟨!utf8⟩ {"\CYRV\cyrery\cyrp\cyru\cyrs\cyrk"}
443       ⟨utf8⟩ {"Выпуск"}
444       { curlanguage "german" =
445         {"{Heft}" } %%% { "Ausgabe" }
446         {"language is not defined: " language "iissue" * * warning$ "Issue"}
447         if$}
448       if$}
449     if$}
450   if$}
451

```

bbl.iiss

```

452 FUNCTION {bbl.iiss} % { "\bblIss{" }
453 { curlanguage "english" =
454   {"Iss."}
455   { curlanguage "ukrainian" =
456     ⟨!utf8⟩ {"\CYRV\cyri\cyrp."}
457     ⟨utf8⟩ {"Вип."}
458     { curlanguage "russian" =
459       ⟨!utf8⟩ {"\CYRV\cyrery\cyrp."}
460       ⟨utf8⟩ {"Вып."}
461       { curlanguage "german" =
462         {"{H.}" }
463         {"language is not defined: " language "iiss" * * warning$ "Iss."}
464         if$}
465       if$}
466     if$}
467   if$}
468

```

bbl.of

```

469 FUNCTION {bbl.of} % { "\bblof{" }
470 { curlanguage "english" =
471   {"of"}
472   { curlanguage "german" =
473     {"von"}
474     { curlanguage "ukrainian" =
475       ⟨!utf8⟩ { "{\cyr\cyrii\cyrz}" }
476       ⟨utf8⟩ { "{із}" }

```



```

477         { curlanguage "russian" =
478 <!utf8>           { "{\cyr\cyri\cyrz}" }
479 <utf8>           { "{\u3}" }
480         {"language is not defined: " language "of" * * warning$ "of"}
481         if$}
482     if$}
483 if$}
484 if$}
485

```

bbl.etal

```

486 FUNCTION {bbl.etal} % { "\bbl of{}" }
487 { curlanguage "english" =
488     {"et~al."}
489     { curlanguage "german" =
490         { "u.~a." }
491         { curlanguage "ukrainian" =
492 <!utf8>           {"{\cyr\cyrt\cyra~\cyrii\cyrn.}" }
493 <utf8>           {"{\ra~iH.}" }
494         { curlanguage "russian" =
495 <!utf8>           {"{\cyr\cyri~\cyrd\cyrr.}" }
496 <utf8>           {"{\u~dp.}" }
497         {"language is not defined: " language "et~al" * * warning$ "et~al."}
498         if$}
499     if$}
500 if$}
501 if$}
502

```

bbl.and

```

503 <natbib>
504 FUNCTION {bbl.and} % { "\bbl and{}" }
505 { curlanguage "english" =
506     {"and"}
507     { curlanguage "german" =
508         { "und" }
509         { curlanguage "ukrainian" =
510 <!utf8>           {"{\cyrii}" }
511 <utf8>           {"i"}
512         { curlanguage "russian" =
513 <!utf8>           {"{\cyri}" }
514 <utf8>           {"\u"}
515         {"language is not defined: " language "and" * * warning$ "and"}
516         if$}
517     if$}
518 if$}
519 if$}
520 </natbib>
521

```

bbl.number

```

522 FUNCTION {bbl.number} % { "\bbl Number{}" }

```

```

523 { curlanguage "english" =
524   {"Number"}
525   { curlanguage "ukrainian" = curlanguage "russian" = or
526     \!utf8 { "\CYRN\cyro\cyrm\cyre\cyrr" }
527     \utf8 { "{Homep}" }
528     { curlanguage "german" =
529       {"{Heft}"} %%% { "Anzahl" }
530       {"language is not defined: " language "nnumber" * * warning$ "Number"}
531       if$}
532   if$}
533 if$}
534

```

bbl.number

```

535 FUNCTION {bbl.number} % { "\bblnumber{" }
536 { curlanguage "english" =
537   {"number"}
538   { curlanguage "ukrainian" = curlanguage "russian" = or
539     \!utf8 { "{\cyr\cyrm\cyro\cyrm\cyre\cyrr}" }
540     \utf8 { "{homep}" }
541     { curlanguage "german" =
542       {"{Heft}"} %%% { "anzahl" }???
543       {"language is not defined: " language "number" * * warning$ "number"}
544       if$}
545   if$}
546 if$}
547

```

bbl.nr

```

548 FUNCTION {bbl.nr} % { "\bblno{" }
549 { curlanguage "english" =
550   {"no."}
551   { curlanguage "italian" =
552     {"no" }
553     { curlanguage "ukrainian" = curlanguage "russian" = or
554       \!utf8 { "{\cyr\textnumero}" }
555       \utf8 { "{#}" }
556       { curlanguage "german" =
557         {"{H.}"} %%% { "an." }
558         {"language is not defined: " language "nr" * * warning$ "no."}
559         if$}
560     if$}
561   if$}
562 if$}
563

```

bbl.nnr

```

564 FUNCTION {bbl.nnr} % { "\bblno{" }
565 { curlanguage "english" =
566   {"No."}
567   { curlanguage "ukrainian" = curlanguage "russian" = or
568     \!utf8 { "{\cyr\textnumero}" }

```

```

569 <utf8>      { "{P}" }
570      { curlanguage "german" =
571        {"{H.}" } %%% { "an." }
572        {"language is not defined: " language "nnr" * * warning$ "No."}
573      if$}
574    if$}
575 if$}
576

```

bbl.in

```

577 FUNCTION {bbl.in} % { "\bblin{" }
578 { curlanguage "english" = curlanguage "german" = or
579   {"in"}
580   { curlanguage "ukrainian" = curlanguage "russian" = or
581     <!utf8>      { "{\cyr\cyrv}" }
582     <utf8>      { "{B}" }
583     {"language is not defined: " language "in" * * warning$ "in"}
584   if$}
585 if$}
586

```

bbl.iin

```

587 FUNCTION {bbl.iin} % { "\bblin{" }
588 { curlanguage "english" = curlanguage "german" = or
589   {"In"}
590   { curlanguage "ukrainian" = curlanguage "russian" = or
591     <!utf8>      { "\CYRV" }
592     <utf8>      { "{B}" }
593     {"language is not defined: " language "iin" * * warning$ "In"}
594   if$}
595 if$}
596

```

bbl.pages

```

597 FUNCTION {bbl.pages} % { "\bblpp." }
598 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
599   {"p." } %%% {"pp."}
600   { curlanguage "ukrainian" = curlanguage "russian" = or
601     <!utf8>      {"{\cyr\cyrs.}" }
602     <utf8>      {"{c.}" }
603     { curlanguage "german" =
604       {"S." } %%% {"s." }
605       {"language is not defined: " language "pages" * * warning$ "p."}
606     if$}
607   if$}
608 if$}
609

```

bbl.page

```

610 FUNCTION {bbl.page} % { "\bblp." }
611 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
612   {"p." }

```

```

613 { curlanguage "ukrainian" = curlanguage "russian" = or
614 <utf8>      {"{\cyr\cyrs.}" }
615 <utf8>      {"{c.}" }
616 { curlanguage "german" =
617     {"S."} %%% { "s." }
618     {"language is not defined: " language "page" * * warning$ "p."}
619     if$}
620 if$}
621 if$}
622

```

bb1.ppages

```

623 FUNCTION {bb1.ppages}%      { "\bb1Pp." }
624 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
625     {"P."} %%% { "Pp." }
626     { curlanguage "ukrainian" = curlanguage "russian" = or
627 <utf8>      {"{\cyr\CYRS.}" }
628 <utf8>      {"{C.}" }
629     { curlanguage "german" =
630         {"S."}
631         {"language is not defined: " language "ppages" * * warning$ "P."}
632         if$}
633     if$}
634 if$}
635

```

bb1.ppage

```

636 FUNCTION {bb1.ppage} %      { "\bb1P." }
637 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
638     {"P."}
639     { curlanguage "ukrainian" = curlanguage "russian" = or
640 <utf8>      {"{\cyr\CYRS.}" }
641 <utf8>      {"{C.}" }
642     { curlanguage "german" =
643         {"S."}
644         {"language is not defined: " language "ppage" * * warning$ "P."}
645         if$}
646     if$}
647 if$}
648

```

bb1.urldate Next function was added in version 2012.01.15.

```

649 FUNCTION {bb1.urldate}
650 { curlanguage "english" =
651     {"online; accessed"}
652     { curlanguage "ukrainian" =
653 <utf8>      { "{\cyrd\cyra\cyrt\cyra\ \cyrz\cyrv\cyre\cyrr\cyrn\cyre\cyrn\cyrn\cyrya}" }
654 <utf8>      { "{дата звернення}" }
655     { curlanguage "russian" =
656 <utf8>      { "{\cyrd\cyra\cyrt\cyra\ \cyro\cyrb\cyrr\cyra\cyrshch\cyre\cyrn\cyri\cyrya}" }
657 <utf8>      { "{дата обращения}" }
658     { curlanguage "german" =

```

```

659         { "{online; abgerufen}" }
660         { "language is not defined: " language "urldate" * * warning$ "online; accessed" }
661         if$}
662     if$}
663 if$}
664 if$}
665

```

bbl.techreport

```

666 FUNCTION {bbl.techreport} % rename to bbl.techreport
667 { curlanguage "english" =
668   { "Rep." }
669   { curlanguage "german" =
670     { "Bericht" }
671     { curlanguage "russian" =
672       { "{\cyr\CYRO\cyrt\cyrch\cyre\cyrt}" }
673       { "{Орчер}" }
674       { "language is not defined: " language "techrep" * * warning$ "Rep." }
675       if$}
676     if$}
677   if$}
678

```

bbl.mthesis

```

679 FUNCTION {bbl.mthesis}
680 { curlanguage "english" =
681   { "Master's thesis" }
682   { curlanguage "german" =
683     { "Diss.~Mag." }
684     { curlanguage "russian" =
685       { "{\cyr\CYRK\cyrv\cyra\cyrl\cyri\cyrf\cyri\cyrk\cyra\cyrc\cyri"
686         { "\cyro\cyrn\cyrn\cyra\cyrya\ \cyrr\cyra\cyrb\cyro\cyrt\cyra\ " *
687         { "\cyrm\cyra\cyrg\cyri\cyrs\cyrt\cyrr\cyra}" * }
688         { "{Квалификационная работа магистра}" }
689         { "language is not defined: " language "mthesis" * * warning$ "Master's thesis" }
690         if$}
691       if$}
692     if$}
693

```

bbl.phdthesis

```

694 FUNCTION {bbl.phdthesis}
695 { curlanguage "english" =
696   { "Ph.\,D. thesis" }
697   { curlanguage "german" =
698     { "Diss.~Ph.\,D." }
699     { curlanguage "russian" =
700       { "{\cyr\CYRD\cyri\cyrs\cyrs\ldots\ \cyrk\cyra\cyrn\cyrd\cyri"
701         { "\cyrd\cyra\cyrt\cyra\ \cyrn\cyra\cyru\cyrk}" * }
702         { "{Дисс\ldots\ кандидата наук}" }
703       { curlanguage "french" =
704         { "Th\{'e}se de doctorat" }

```

```

705         { "language is not defined: " language "phdthesis" * * warning$ "Ph.\,D. thesis" }
706     if$}
707 if$}
708 if$}
709 if$}
710

```

bb1.dscithesis

```

711 FUNCTION {bb1.dscithesis}
712 { curlanguage "english" =
713   { "Dr.\,Sci. dissertation" }
714   { curlanguage "german" =
715     { "Diss.~Dr." }
716     { curlanguage "russian" =
717       { "\cyr\CYRD\cyri\cyrs\cyrs\ldots\ \cyrd\cyro\cyrk\cyrt\cyro"
718         "\cyrr\cyra\ \cyrn\cyra\cyru\cyrk}" * }
719       { "\Дисс\ldots\ доктора наук}" }
720       { "language is not defined: " language "dscithesis" * * warning$ "Dr.\,Sci. dissertation" }
721     if$}
722   if$}
723 if$}
724

```

bb1.nnoaddress

```

725 FUNCTION {bb1.nnoaddress}
726 { curlanguage "english" =
727   { "S.\ 1." }
728   { curlanguage "russian" =
729     { "\cyr\CYRB.\ \cyrm.}" }
730     { "\Б.\ М.}" }
731     { "language is not defined: " language "nnoaddress" * * warning$ "S.\ 1." }
732   if$}
733 if$}
734

```

bb1.nopublisher

```

735 FUNCTION {bb1.nopublisher}
736 { curlanguage "english" =
737   { "s.\ n." }
738   { curlanguage "russian" =
739     { "\cyr\cyrb.\ \cyri.}" }
740     { "\б.\ н.}" }
741     { "language is not defined: " language "nopublisher" * * warning$ "s.\ n." }
742   if$}
743 if$}
744

```

bb1.nnopublisher

```

745 FUNCTION {bb1.nnopublisher}
746 { curlanguage "english" =
747   { "S.\ n." }
748   { curlanguage "russian" =

```

```

749 <utf8>      { "{\cyr\CYRB.\ \cyri.}" }
750 <utf8>      { "{Б.\ И.}" }
751      { "language is not defined: " language "nnopublisher" * * warning$ "S.\ n." }
752      if$}
753 if$}
754

```

bbl.medium.text

```

755 FUNCTION {bbl.medium.text}
756 { curlanguage "english" =
757   { "Text" }
758   { curlanguage "russian" = curlanguage "ukrainian" = or
759 <utf8>      { "{\cyr\CYRT\cyre\cyrk\cyrs\cyrt}" }
760 <utf8>      { "{Текст}" }
761   { "language is not defined: " language "medium" * * warning$ "Text" }
762   if$}
763 if$}
764

```

bbl.medium.elres

```

765 FUNCTION {bbl.medium.elres}
766 { curlanguage "english" =
767   { "Electronic resource" }
768   { curlanguage "russian" =
769 <utf8>      { "{\cyr\CYREREV\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyrery\cyrishrt\ \cyrr\cyre\cyrs\cyru\cyrr}" }
770 <utf8>      { "{Электронный ресурс}" }
771   { curlanguage "ukrainian" =
772 <utf8>      { "{\cyr\CYRE\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyri\cyrishrt\ \cyrr\cyre\cyrs\cyru\cyrr}" }
773 <utf8>      { "{Електронний ресурс}" }
774   { "language is not defined: " language "medium" * * warning$ "Electronic resource" }
775   if$}
776   if$}
777 if$}
778
779

```

bbl.chief

```

780 FUNCTION {bbl.chief}
781 { curlanguage "english" =
782   { "chief" }
783   { curlanguage "russian" =
784 <utf8>      { "\cyrr\cyru\cyrk." }
785 <utf8>      { "{пук.}" }
786   { curlanguage "ukrainian" =
787 <utf8>      { "\cyrr\cyru\cyrk." }
788 <utf8>      { "{пук.}" }
789   { "language is not defined: " language "chief" * * warning$ "chief" }
790   if$}
791   if$}
792 if$}
793

```

bbl.executor

```
794 FUNCTION {bbl.executor}
795 { curlanguage "english" =
796   { "Executor" }
797   { curlanguage "russian" =
798     { "\cyr\cyri\cyrs\cyrp\cyro\cyr1\cyrn." } }
799     { "\utf8" { "{исполн.}" } }
800     { curlanguage "ukrainian" =
801       { "\cyr\cyrv\cyri\cyrk\cyro\cyrn\cyra\cyrv\cyre\cyrc\cyrsfts" } }
802       { "\utf8" { "{виконавець}" } }
803       { "language is not defined: " language "medium" * * warning$ "Executor" }
804       if$}
805   if$}
806 if$}
807
```

bbl.medium

```
808 FUNCTION {bbl.medium}
809 { type$ "online" =
810   { bbl.medium.elres }
811   { bbl.medium.text }
812 if$}
813
```

## 2.4 Formatting functions

Declare functions to format separate elements of bibliographic reference.

```
814 INTEGERS { nameptr namesleft numnames }
815
816
```

format.names

Function `format.names` has 2 version. First is for bibliographic records rather than for bibliographic references. It is used for `.bst` styles compiled without the `modern` option. It formats every name as 'LastName, F. S.'. Historically, this version was used first for earlier styles included into GOST bundle.

### Important note

Neither `bibtex` nor `bibtex8` can handle unencoded text without troubles. In particular, they fail to reduce a Cyrillic name to initials. Therefore we avoid using `f.` primitive when option `utf8` is in effect.

```
817 <!*modern>
818 FUNCTION {format.names}
819 {
820   {!\utf8} #1 "{vv~}{ll}{~jj}{,~f.}" format.name$
821   {!\utf8} #1 "{vv~}{ll}{~jj}{,~ff}" format.name$
822 }
823 </!*modern>
```

Second version drops comma from output so that every name is formatted as 'LastName F. S.'. It also substitutes 4th and following names by localized term 'et al'.

```
824 <!*modern>
```



```

825 FUNCTION {format.names}
826 {
827   's :=
828   #1 'nameptr :=
829   s num.names$ 'numnames :=
830   numnames 'namesleft :=
831   { namesleft #0 > }
832   { s nameptr
833     (!utf8)      "{vv~}{ll}{~jj}{~f.}" format.name$ 't :=
834     (utf8)       "{vv~}{ll}{~jj}{~ff}" format.name$ 't :=
835     nameptr #1 >
836     { nameptr #4 =
837       numnames #4 > and
838       { "others" 't :=
839         #1 'namesleft := }
840       'skip$
841       if$
842       namesleft #1 >
843       { ", " * t * }
844       { t "others" =
845         t "~others" =
846         or
847         (!strict)      { " " * bbl.etal * }
848         (strict)       { " " * bbl.etal enclose.square.brackets *}
849         { ", " * t * }
850         if$
851         }
852         if$
853         }
854         't
855         if$
856         nameptr #1 + 'nameptr :=
857         namesleft #1 - 'namesleft :=
858         }
859     while$
860   }
861 }
862

```

`format.lab.names`    Declare function to go to optional argument of `\bibitem` in the styles generated with the option `natbib`.

```

863 (*natbib)
864 FUNCTION {format.lab.names}
865 { 's :=
866   language empty$
867   { "english" 'curlanguage := }
868   { language 'curlanguage := }
869   if$
870   s #1 "{vv~}{ll}" format.name$
871   s num.names$ duplicate$
872   #2 >

```

```

873     %{ pop$ " et~al." * }
874     { pop$ " " bbl.etal * * }
875     { #2 <
876         'skip$
877         { s #2 "{ff }{vv }{ll}{ jj}" format.name$ "others" =
878             %{ " et~al." * }
879             { " " bbl.etal * * }
880             %{ " and " * s #2 "{vv~}{ll}" format.name$ * }
881             { " " bbl.and " " * * * s #2 "{vv~}{ll}" format.name$ * }
882         if$
883     }
884     if$
885 }
886 if$
887 }
888 </natbib>
889

```

**format.names.rev** Declare function to format names for authors/bookauthors list after title and etc. Note that **format.names.rev** cuts list of names to at most 4 persons. We do not cut names to initials in this list.

```

890 FUNCTION {format.names.rev}
891 {
892     's :=
893     #1 'nameptr :=
894     s num.names$ 'numnames :=
895     numnames 'namesleft :=
896     { namesleft #0 > }
897     { s nameptr
898     <!utf8>         %"{f.}{~vv}{~ll}{, jj}" format.name$ 't :=
899     <!utf8>         "%{ff}{~vv}{~ll}{, jj}" format.name$ 't :=
900     <utf8>         "%{ff}{~vv}{~ll}{, jj}" format.name$ 't :=
901     nameptr #1 >
902     { nameptr #4 =
903         numnames #4 > and
904         { "others" 't :=
905             #1 'namesleft := }
906         'skip$
907     if$
908     namesleft #1 >
909     { ", " * t * }
910     { t "others" =
911     t "~others" =
912     or
913     <!strict>         { " " * bbl.etal * }
914     <strict>         { " " * bbl.etal enclose.square.brackets * }
915     { ", " * t * }
916     if$
917     }
918     if$
919 }

```

```

920      't
921      if$
922      nameptr #1 + 'nameptr :=
923      namesleft #1 - 'namesleft :=
924    }
925    while$
926 }
927

```

`format.key`    Function to substitute empty field (usually, author name) with the key field.

```

928 (*natbib)
929 FUNCTION {format.key}
930 { empty$
931   { key field.or.null }
932   { "" }
933   if$
934 }
935 (/natbib)
936

```

`format.authors`

```

937 FUNCTION {format.authors}
938 { author empty$
939   { "" }
940   { author format.names emphasize}
941   if$
942 }
943

```

`author.key.label`

```

944 (*natbib)
945 FUNCTION {author.key.label}
946 { author empty$
947   { key empty$
948     { cite$ #1 #3 substring$ }
949     'key
950     if$
951   }
952   { author format.lab.names }
953   if$
954 }
955
956 FUNCTION {author.editor.key.label}
957 { author empty$
958   { editor empty$
959     { key empty$
960       { cite$ #1 #3 substring$ }
961       %'key          %% causes lost of year
962       { "{}" key * } %% Bug in bibtex8 ??
963       if$
964     }
965     { editor format.lab.names }

```

```

966         if$
967     }
968     { author format.lab.names }
969     if$
970 }
971
972 FUNCTION {author.key.organization.label}
973 { author empty$
974     { key empty$
975         { organization empty$
976             { cite$ #1 #3 substring$ }
977             { "The " #4 organization chop.word #3 text.prefix$ }
978             if$
979         }
980         'key
981         if$
982     }
983     { author format.lab.names }
984     if$
985 }
986
987 FUNCTION {editor.key.organization.label}
988 { editor empty$
989     { key empty$
990         { organization empty$
991             { cite$ #1 #3 substring$ }
992             { "The " #4 organization chop.word #3 text.prefix$ }
993             if$
994         }
995         'key
996         if$
997     }
998     { editor format.lab.names }
999     if$
1000 }
1001
1002 FUNCTION {calc.short.authors}
1003 { type$ "book" =
1004     type$ "inbook" =
1005     or
1006     'author.editor.key.label
1007     { type$ "proceedings" =
1008         'editor.key.organization.label
1009         { type$ "manual" =
1010             'author.key.organization.label
1011             'author.key.label
1012             if$
1013         }
1014         if$
1015     }

```

```

1016   if$
1017   'short.list :=
1018 }
1019
1020 FUNCTION {calc.label}
1021 { calc.short.authors
1022   short.list
1023   "("
1024   *
1025   year duplicate$ empty$
1026   short.list key field.or.null = or
1027     { pop$ "" }
1028     'skip$
1029   if$
1030   *
1031   'label :=
1032 }
1033
1034 </natbib>
1035

```

format.bookauthors This function is used only once, in bookauthor.before, and the latter is used only in inbook entry.

```

1036 FUNCTION {format.bookauthors}
1037 { bookauthor empty$
1038   { "" }
1039   { bookauthor format.names}
1040   if$
1041 }
1042

```

mat.authors.after

```

1043 FUNCTION {format.authors.after}
1044 { author empty$
1045   { "" }
1046   { author format.names.rev}
1047   if$
1048 }
1049

```

bookauthors.after

```

1050 FUNCTION {format.bookauthors.after}
1051 { bookauthor empty$
1052   { "" }
1053   { bookauthor format.names.rev}% always cuts to 4 persons
1054   if$
1055 }
1056

```

mat.editors.after

```

1057 FUNCTION {format.editors.after}
1058 { editor empty$

```

```

1059     { "" }
1060     { bbl.edby "\ " * editor format.names.rev * }
1061   if$
1062 }
1063
format.chief.after
1064 FUNCTION {format.chief.after}
1065 { editor empty$
1066   { "" }
1067   { bbl.chief "\ " * editor format.names.rev * }
1068   if$
1069 }
1070
at.executor.after
1071 FUNCTION {format.executor.after}
1072 { author empty$
1073   { "" }
1074   { bbl.executor ": " * author format.names.rev * }
1075   if$
1076 }
1077
at.compiler.after
1078 FUNCTION {format.compiler.after}
1079 { compiler empty$
1080   { "" }
1081   { bbl.cmplr "\ " * compiler format.names.rev * }
1082   if$
1083 }
1084
format.title   Important note Neither bibtex nor bibtex8 can handle unencoded text without troubles. In
                particular, bibtex8 fails to change case of a string if it contains Cyrillic letter. Therefore we avoid
                using change.case$ when option utf8 is in effect.
1085 FUNCTION {format.title}
1086 { title empty$
1087   { "" }
1088   <utf8> { title "t" change.case$ }
1089   <utf8> { title }
1090   if$
1091 }
1092
format.date
1093 FUNCTION {format.date}
1094 { year empty$
1095   { month empty$
1096     { "" }
1097     { "there's a month but no year in " cite$ * warning$
1098       month

```

```

1099     }
1100     if$
1101   }
1102   { month empty$
1103     'year
1104 <!modern>      { year ". " quote$ "--- " month * * * * }
1105 <modern>      { year ". \BibDash " month * * }
1106     if$
1107   }
1108   if$
1109 <natbib> extra.label * % new in v.1.2
1110 }
1111

```

address.publisher

address.publisher.date

```

1112 <!*strict>
1113 FUNCTION {output.address.publisher}
1114 {
1115   address empty$
1116     'skip$
1117   { address output
1118     publisher empty$
1119       'skip$
1120       { new.column }
1121     if$
1122   }
1123   if$
1124   publisher output
1125 }
1126 </!strict>
1127 <*strict>
1128 %FUNCTION {output.address.publisher}
1129 %{
1130 %   address empty$
1131 %   {
1132 %     bbl.nnoaddress
1133 %     publisher empty$
1134 %     { "~: " * bbl.nopublisher * }
1135 %     { }
1136 %     if$
1137 %     enclose.square.brackets
1138 %   }
1139 %   {
1140 %     address output
1141 %     new.column
1142 %     publisher empty$
1143 %     { bbl.nopublisher enclose.square.brackets }
1144 %     { publisher }
1145 %     if$
1146 %   }
1147 %   if$

```

```

1148 % output
1149 %}
1150 FUNCTION {output.address.publisher}
1151 {
1152   address empty$
1153   {
1154     bbl.nnoaddress
1155     publisher empty$
1156     { "~: " * bbl.nopublisher * enclose.square.brackets }
1157     { enclose.square.brackets "~: " * publisher * }
1158     if$
1159   }
1160   {
1161     address output
1162     new.column
1163     publisher empty$
1164     { bbl.nopublisher enclose.square.brackets }
1165     { publisher }
1166     if$
1167   }
1168   if$
1169   output
1170 }
1171
1172 </strict>
1173

```

ress.publisher.date Otput.address.publisher.date is used in old styles. New styles use output.address.publisher.

```

1174 <*(modern | strict)>
1175 FUNCTION {output.address.publisher.date}
1176 {
1177   output.address.publisher
1178   format.date output
1179 }
1180 </!(modern | strict)>
1181

```

output.bibitem

```

1182 <*(natbib)>
1183 FUNCTION {output.bibitem}
1184 { newline$
1185   "\bibitem" write$
1186   cite$ braceify write$
1187   newline$
1188   language empty$
1189   { "english" 'curlanguage := }
1190   { language 'curlanguage := }
1191   if$
1192   "\selectlanguageifdefined" curlanguage braceify * write$
1193   newline$
1194   ""

```



```

1195 before.all 'output.state :=
1196 }
1197 </!natbib>

format.full.names      In case of natbib option, we need make.full.names to compose output.bibitem, and the
author.full            latter in its turn requires some more functions.
editor.full            1198 <natbib>
author.editor.full     1199 FUNCTION {format.full.names}
make.full.names        1200 { 's :=
output.bibitem         1201   language empty$
                        1202   { "english" 'curlanguage := }
                        1203   { language 'curlanguage := }
                        1204   if$
                        1205   #1 'nameptr :=
                        1206   s num.names$ 'numnames :=
                        1207   numnames 'namesleft :=
                        1208   { namesleft #0 > }
                        1209   { s nameptr
                        1210     "{vv~}{ll}" format.name$ 't :=
                        1211     nameptr #1 >
                        1212     {
                        1213       namesleft #1 >
                        1214       { ", " * t * }
                        1215       {
                        1216         numnames #2 >
                        1217         curlanguage "english" =
                        1218         and
                        1219         { ", " * }
                        1220         'skip$
                        1221         if$
                        1222         t "others" =
                        1223         %t "~others" =
                        1224         %or
                        1225         %{ " et~al." * }
                        1226         { " " bbl.etal * * }
                        1227         %{ " and " * t * }
                        1228         { " " bbl.and " " * * * t * }
                        1229         if$
                        1230       }
                        1231       if$
                        1232     }
                        1233     't
                        1234     if$
                        1235     nameptr #1 + 'nameptr :=
                        1236     namesleft #1 - 'namesleft :=
                        1237   }
                        1238   while$
                        1239 }
1240
1241 FUNCTION {author.full}
1242 { author empty$

```

```

1243     { "" }
1244     { author format.full.names }
1245     if$
1246 }
1247
1248 FUNCTION {editor.full}
1249 { editor empty$
1250     { "" }
1251     { editor format.full.names }
1252     if$
1253 }
1254
1255 FUNCTION {author.editor.full}
1256 { author empty$
1257     { editor empty$
1258         { "" }
1259         { editor format.full.names }
1260         if$
1261     }
1262     { author format.full.names }
1263     if$
1264 }
1265
1266 FUNCTION {make.full.names}
1267 { type$ "book" =
1268     type$ "inbook" =
1269     or
1270     'author.editor.full
1271     { type$ "proceedings" =
1272         'editor.full
1273         'author.full
1274         if$
1275     }
1276     if$
1277 }
1278
1279 % =====
1280 FUNCTION {output.bibitem}
1281 { newline$
1282     "\bibitem[" write$
1283     label write$
1284     ")" make.full.names duplicate$ short.list =
1285         { pop$ }
1286         { * }
1287         if$
1288     "]" * write$
1289     cite$ write$
1290     "]" write$
1291     language empty$
1292     { "english" 'curlanguage := }

```

```

1293     {language 'curlanguage := }
1294   if$
1295   "\selectlanguageifdefined" curlanguage braceify * write$
1296   newline$
1297   ""
1298   before.all 'output.state :=
1299 }
1300 % =====
1301 %FUNCTION {output.bibitem}
1302 %{ newline$
1303 % "\bibitem" write$
1304 %% author.key.label
1305 %% year parenthesify *
1306 %% "; lbl:" label * *
1307 %% "; mfn:" make.full.names * *
1308 % label
1309 % make.full.names *
1310 % bracketise write$
1311 % cite$ braceify write$
1312 % newline$
1313 % language empty$
1314 % { "english" 'curlanguage := }
1315 % {language 'curlanguage := }
1316 % if$
1317 % "\selectlanguageifdefined" curlanguage braceify * write$
1318 % newline$
1319 % ""
1320 % before.all 'output.state :=
1321 %}
1322 % =====
1323 </natbib>
1324

```

n.dashify

```

1325 FUNCTION {n.dashify}
1326 { 't :=
1327   ""
1328   { t empty$ not }
1329   { t #1 #1 substring$ "-" =
1330     { t #1 #2 substring$ "--" = not
1331       { "--" *
1332         t #2 global.max$ substring$ 't :=
1333       }
1334       { { t #1 #1 substring$ "-" = }
1335         { "-" *
1336           t #2 global.max$ substring$ 't :=
1337         }
1338       while$
1339     }
1340     if$
1341   }

```

```

1342      { t #1 #1 substring$ *
1343      t #2 global.max$ substring$ 't :=
1344      }
1345      if$
1346      }
1347      while$
1348 }
1349

word.in
1350 FUNCTION {word.in}
1351 { bbl.iin
1352   " " * }
1353

format.btitle
1354 FUNCTION {format.btitle}
1355 { title
1356 }
1357

tie.or.space.connect
1358 FUNCTION {tie.or.space.connect}
1359 { duplicate$ text.length$ #3 <
1360   { "~" }
1361   { " " }
1362   if$
1363   swap$ * *
1364 }
1365

tie.connect  Declare function to insert unbreakable space before last word in stack.
1366 FUNCTION {tie.connect}
1367 {"~"
1368   swap$ * *
1369 }
1370
1371

either.or.chec
1372 FUNCTION {either.or.check}
1373 { empty$
1374   'pop$
1375   { "can't use both " swap$ * " fields in " * cite$ * warning$ }
1376   if$
1377 }
1378

format.bvolume
1379 FUNCTION {format.bvolume}
1380 { volume empty$
1381   { "" }
1382   { bbl.vvol volume tie.connect

```

```

1383     series empty$
1384     'skip$
1385     { bbl.of space.word * series emphasize * }
1386     if$
1387     "volume and number" number either.or.check
1388   }
1389   if$
1390 }
1391

```

mat.number.series

```

1392 FUNCTION {format.number.series}
1393 { volume empty$
1394   { number empty$
1395     { series field.or.null }
1396     { series empty$
1397       { "there's a number but no series in " cite$ * warning$
1398         bbl.nnr }
1399       {
1400         %new.dblslash
1401         new.sentence
1402         series
1403         bbl.nr
1404         tie.or.space.connect}
1405       if$
1406       number tie.or.space.connect
1407     }
1408     if$
1409   }
1410   { "" }
1411   if$
1412 }
1413

```

is.num

```

1414 FUNCTION {is.num}
1415 { chr.to.int$
1416   duplicate$ "0" chr.to.int$ < not
1417   swap$ "9" chr.to.int$ > not and
1418 }
1419

```

extract.num

```

1420 FUNCTION {extract.num}
1421 { duplicate$ 't :=
1422   "" 's :=
1423   { t empty$ not }
1424   { t #1 #1 substring$
1425     t #2 global.max$ substring$ 't :=
1426     duplicate$ is.num
1427     { s swap$ * 's := }
1428     { pop$ "" 't := }

```

```

1429     if$
1430   }
1431   while$
1432     s empty$
1433     'skip$
1434     { pop$ s }
1435   if$
1436 }
1437
1438 <debug>

```

eng.ord

```

1439 FUNCTION {eng.ord}
1440 { duplicate$ "1" swap$ *
1441   #-2 #1 substring$ "1" =
1442     { bbl.th * }
1443   { duplicate$ #-1 #1 substring$
1444     duplicate$ "1" =
1445       { pop$ bbl.st * }
1446     { duplicate$ "2" =
1447       { pop$ bbl.nd * }
1448     { "3" =
1449       { bbl.rd * }
1450     { bbl.th * }
1451     if$
1452   }
1453   if$
1454 }
1455   if$
1456 }
1457   if$
1458 }
1459 </debug>
1460

```

convert.edition

```

1461 FUNCTION {convert.edition}
1462 { edition
1463 % edition extract.num "1" change.case$ 's :=
1464 %   s "first" = s "1" = or
1465 %   { bbl.first 't := }
1466 %   { s "second" = s "2" = or
1467 %   { bbl.second 't := }
1468 %   { s "third" = s "3" = or
1469 %   { bbl.third 't := }
1470 %   { s "fourth" = s "4" = or
1471 %   { bbl.fourth 't := }
1472 %   { s "fifth" = s "5" = or
1473 %   { bbl.fifth 't := }
1474 %   { s #1 #1 substring$ is.num
1475 %   { s eng.ord 't := }

```

```

1476 %           { edition 't := }
1477 %           if$
1478 %           }
1479 %           if$
1480 %           }
1481 %           if$
1482 %           }
1483 %           if$
1484 %           }
1485 %           if$
1486 %           }
1487 % if$
1488 % t
1489 }
1490

```

format.edition

```

1491 FUNCTION {format.edition}
1492 { edition empty$
1493   { "" }
1494   { output.state mid.sentence =
1495     { \utf8 }      { convert.edition "l" change.case$ " " * bbl.edition * }
1496     { \utf8 }      { convert.edition "t" change.case$ " " * bbl.edition * }
1497     { \utf8 }      { convert.edition " " * bbl.edition * }
1498     { \utf8 }      { convert.edition " " * bbl.edition * }
1499     if$
1500   }
1501   if$
1502 }
1503
1504 INTEGERS { multiresult }
1505

```

multi.page.check

```

1506 FUNCTION {multi.page.check}
1507 { 't :=
1508   #0 'multiresult :=
1509   { multiresult not
1510     t empty$ not
1511     and
1512   }
1513   { t #1 #1 substring$
1514     duplicate$ "-" =
1515     swap$ duplicate$ "," =
1516     swap$ "+" =
1517     or or
1518     { #1 'multiresult := }
1519     { t #2 global.max$ substring$ 't := }
1520     if$
1521   }
1522   while$

```

```

1523 multiresult
1524 }
1525

```

format.pages

```

1526 FUNCTION {format.pages}
1527 { pages empty$
1528   { "" }
1529   { pages multi.page.check
1530     { bbl.ppages pages n.dashify tie.connect }
1531     { bbl.ppage pages tie.connect }
1532     if$
1533   }
1534   if$
1535 }
1536

```

format.pages.page

```

1537 FUNCTION {format.pages.page}
1538 { pages empty$
1539   { numpages empty$
1540     { "" }
1541     { numpages bbl.pages tie.connect }
1542     if$}
1543   { format.pages}
1544   if$
1545 }
1546

```

mat.vol.num.pages

```

1547 FUNCTION {format.vol.num.pages}
1548 { volume field.or.null
1549   number empty$
1550   'skip$
1551   {
1552     ", no." number tie.or.space.connect *
1553     volume empty$
1554     { "there's a number but no volume in " cite$ * warning$ }
1555     'skip$
1556     if$
1557   }
1558   if$
1559   pages empty$
1560   'skip$
1561   { duplicate$ empty$
1562     { pop$ format.pages }
1563     { ": " * pages n.dashify * }
1564     if$
1565   }
1566   if$
1567 }
1568

```



```

format.volume
1569 FUNCTION {format.volume}
1570 { volume empty$
1571   { "" }
1572   { bbl.vvol volume tie.or.space.connect }
1573   if$
1574 }
1575

format.number
1576 FUNCTION {format.number}
1577 { number empty$
1578   { "" }
1579   { bbl.nr number tie.or.space.connect }
1580   if$
1581 }
1582
1583 <*debug>

mat.chapter.pages
1584 FUNCTION {format.chapter.pages}
1585 { chapter empty$
1586   'format.pages
1587   { type empty$
1588     { bbl.chapter }
1589     { type "1" change.case$ }
1590     if$
1591     chapter tie.or.space.connect
1592     pages empty$
1593     'skip$
1594     { ", " * format.pages * }
1595     if$
1596   }
1597   if$
1598 }
1599 </debug>
1600

empty.misc.check
1601 FUNCTION {empty.misc.check}
1602 { author empty$ title empty$ howpublished empty$
1603   month empty$ year empty$ note empty$
1604   and and and and and
1605   key empty$ not and
1606   { "all relevant fields are empty in " cite$ * warning$ }
1607   'skip$
1608   if$
1609 }
1610

ormat.thesis.type
1611 FUNCTION {format.thesis.type}

```

```

1612 { type empty$
1613   'skip$
1614   { pop$
1615     <!utf8>      type "t" change.case$
1616     <utf8>       type
1617   }
1618   if$
1619 }
1620

```

**chrep.type.number**    Function to format report type and number.

```

1621 %FUNCTION {format.techrep.type.number}
1622 %{ type empty$
1623 %   { bbl.techreport }
1624 %   'type
1625 %   if$
1626 %   number empty$
1627 %<!utf8>   { "t" change.case$ }
1628 %<utf8>    { "" }
1629 %   { number tie.or.space.connect }
1630 %   if$
1631 %}
1632
1633 FUNCTION {format.techreport.type}
1634 { type empty$
1635   { bbl.techreport }
1636   'type
1637   if$
1638 }
1639

```

**author.before**    Declare the function **author.before** to format list of authors in heading of a bibliographic record. If the number of authors is 4 or larger, some styles skip the list of authors in the beginning of the bibliographic record, while other styles always print that list. So, we need two version of **author.before**.

First version is used if **.bst** style is compiled without option **long**. It skips authors if their number is greater than or equal to 4 or if the author field is empty. Note that GOST requires for a long list of authors to be reduced. Hence this first version is preferable. Note also that **format.names** cuts list of names to 4 person at most in case if **modern** option is used. and **format.authors** just emphasizes **format.names**.

```

1640 <!*long>
1641 FUNCTION {author.before}
1642 {
1643   author empty$
1644   'skip$
1645   {author num.names$ #4 <
1646     {format.authors output
1647       new.sentence}
1648     'skip$
1649     if$}

```

```

1650  if$
1651 }
1652 </!long>

```

Second version is used if `.bst` style is compiled with the option `long`. It skips only if the author field is empty.

```

1653 <!*long>
1654 FUNCTION {author.before}
1655 {
1656   author empty$
1657   'skip$
1658   { format.authors output
1659     new.sentence
1660   }
1661   if$
1662 }
1663 </long>
1664

```

`bookauthor.before` There are also 2 version of the function `bookauthor.before`. Not used anymore!

```

1665 %%<!*long>
1666 %%FUNCTION {bookauthor.before}
1667 %%{
1668 %%  bookauthor empty$
1669 %%    'skip$
1670 %%    {bookauthor num.names$ #4 <
1671 %%      {format.bookauthors output
1672 %%        new.sentence}
1673 %%      'skip$
1674 %%    if$}
1675 %%  if$
1676 %%}
1677 %%</!long>
1678 %%<!*long>
1679 %%FUNCTION {bookauthor.before}
1680 %%{
1681 %%  bookauthor empty$
1682 %%    'skip$
1683 %%    { format.bookauthors output
1684 %%      new.sentence
1685 %%    }
1686 %%  if$
1687 %%}
1688 %%</long>
1689

```

`author.after` Functions `author.after` and `bookauthor.after` also have by 2 versions. They are used to write authors list after the title followed by a slash. In modern styles, compiled with option `modern`, the list of authors is always cut to at most 4 persons. The cut is performed first by `format.names.rev`, which is called by `format.authors.after`. For old styles, `author.after` just outputs formatted string whereas for new style it skips the string if the number of authors exceeds 3 (and author list

is not printed before the title).

```
1690 <!*modern>
1691 FUNCTION {author.after}
1692 {
1693   author empty$
1694   'skip$
1695   {format.authors.after output
1696     new.semicolumn }
1697   if$
1698 }
1699 </!modern>
1700 <*modern>
1701 <!*long>
1702 FUNCTION {author.after}
1703 {
1704   author empty$
1705   'skip$
1706   {author num.names$ #3 >
1707     {format.authors.after output
1708       new.semicolumn }
1709     'skip$
1710     if$}
1711   if$
1712 }
1713 </!long>
1714 <*long>
1715 FUNCTION {author.after} { }
1716 </long>
1717 </modern>
1718
```

`bookauthor.after` This function is used only in `inbook` entry. It always cuts list to 4 persons since `format.bookauthors.after` does that.

```
1719 FUNCTION {bookauthor.after}
1720 {
1721   bookauthor empty$
1722   'skip$
1723   {format.bookauthors.after output
1724     new.semicolumn }
1725   if$
1726 }
1727
```

`organization.after`

```
1728 FUNCTION {editor.organization.after}
1729 {
1730   compiler empty$
1731   {}
1732   { format.compiler.after output
1733     new.semicolumn
1734   }
```

```

1735 if$
1736 editor empty$
1737 {}
1738 { format.editors.after output
1739 new.semicolumn
1740 }
1741 if$
1742 organization empty$
1743 {}
1744 {organization output
1745 new.semicolumn
1746 }
1747 if$
1748 }
1749

```

format.url

```

1750 FUNCTION {format.url}
1751 { url empty$
1752   { "" }
1753   {
1754     !(modern|strict)      "\BibUrl{ " url * "}" *
1755     (modern|strict)      "URL: \BibUrl{" url * "}" *
1756     urldate empty$
1757     { "" }
1758     { " (" bbl.urldate * ": " * urldate * ")" * }
1759     if$ *
1760   }
1761   if$
1762 }
1763

```

format.annotate

```

1764 FUNCTION {format.annotate}
1765 { annotate empty$
1766   { "" }
1767   { after.sentence 'output.state :=
1768     "\BibAnnote{" annotate add.period$ * "}" *
1769   }
1770   if$
1771 }
1772

```

format.isbn Do we really need to provide electronic search for ISBN?

```

1773 FUNCTION {format.isbn}
1774 {
1775   isbn empty$
1776   { "" }
1777   { "ISBN:~\href{http://isbndb.com/search-all.html?kw=" isbn *
1778     "}{ " * isbn * "}" *
1779   }
1780   if$

```

```
1781 }
1782
```

**add.doi** The Digital Object Identifier (DOI) System is for identifying content objects in the digital environment. DOI names are assigned to any entity for use on digital networks. They are used to provide current information, including where they (or information about them) can be found on the Internet. Information about a digital object may change over time, including where to find it, but its DOI name will not change.

Function **add.doi** embraces last string in stack into hyperlink that links it to specified doi identificator at <http://dx.doi.org/> web-site.

```
1783 <*eprint>
1784 FUNCTION {add.doi}
1785 { duplicate$ empty$
1786   'skip$
1787   { doi empty$
1788     'skip$
1789     { "\href{http://dx.doi.org/" doi * "}{ " * swap$ * "}" * }
1790     if$
1791   }
1792   if$
1793 }
1794 </eprint>
```

If .bst style is compiled without **eprint** option, we just ignore doi field.

```
1795 <!*eprint>
1796 FUNCTION {add.doi} { }
1797 </!eprint>
1798
```

**add.medium** New in version 2.

```
1799 <!*strict>
1800 FUNCTION {add.medium} { }
1801 </!strict>
1802 <*strict>
1803 FUNCTION {add.medium}
1804 { duplicate$ empty$
1805   'skip$
1806   { medium empty$
1807     { " " * bbl.medium enclose.square.brackets * }
1808     { " " * medium enclose.square.brackets * }
1809     %% { bbl.medium enclose.square.brackets * }
1810     %% { medium enclose.square.brackets * }
1811     if$
1812   }
1813   if$
1814 }
1815 </strict>
1816
```

## 2.5 Electronic Publishing Information

The biblatex package provides three fields for electronic publishing information: `eprint`, `eprinttype`, and `eprintclass`. The `eprint` field is a verbatim field similar to `doi` which holds the identifier of the item. The `eprinttype` field holds the resource name, i. e., the name of the site or electronic archive. The optional `eprintclass` field is intended for additional information specific to the resource indicated by the `eprinttype` field. This could be a section, a path, classification information, etc. If the `eprinttype` field is available, the standard styles will use it as a literal label. In the following example, they would print “Resource: identifier” rather than the generic “eprint: identifier”:

```
eprint = {identifier},
eprinttype = {Resource},
```

`format.eprint`      The electronic identifier of an online publication. This is roughly comparable to a `doi` but specific to a certain archive, repository, service, or system. Also see `eprinttype` and `eprintclass`.  
This function should use `url`. TO BE DONE YET.

```
1817 <*eprint>
1818 %FUNCTION {format.eprint}
1819 %{ eprint empty$
1820 %   { "" }
1821 %   { eprintclass empty$
1822 %     { " \href{http://arxiv.org/abs/" eprint * "}" * "{" * eprint * "}" * }
1823 %     { eprinttype empty$
1824 %       { " \href{http://arxiv.org/abs/" eprint * "}" *
1825 %         {" * eprintclass * "/" * eprint * "}" *
1826 %       }
1827 %       { " \href{http://arxiv.org/abs/" eprint * "}" *
1828 %         {" * eprinttype * ":" * eprintclass * "/" * eprint * "}" *
1829 %       }
1830 %     if$}
1831 %   if$}
1832 %if$}
1833
1834 %FUNCTION {format.eprint}
1835 %{ eprint empty$
1836 %   { "" }
1837 %   { eprinttype empty$
1838 %     { "" }
1839 %     { eprinttype "~: " *}
1840 %   if$
1841 %     eprintclass empty$
1842 %     { }
1843 %     { eprintclass * "/" *}
1844 %   if$
1845 %     eprint *
1846 %   }
1847 % if$
1848 % url empty$
1849 %   { }
```

```

1850 % { "\href{" url * "}{ " * swap$ * "}" *}
1851 % if$
1852 %}
1853
1854 FUNCTION {format.eprint}
1855 { eprint empty$
1856   { "" }
1857   { eprinttype empty$
1858     { "" }
1859     { eprinttype "~: " *}
1860     if$
1861     eprintclass empty$
1862     { }
1863     { eprintclass * "/" *}
1864     if$
1865     url empty$
1866     { eprint * }
1867     { "\href{" * url * "}{ " * eprint * "}" *}
1868     if$
1869   }
1870   if$
1871 }
1872
1873 FUNCTION {output.eprint.url}
1874 {
1875   eprint empty$
1876   { format.url output }
1877   { format.eprint output }
1878   if$
1879 }
1880
1881 </eprint>
1882
1883 <!*eprint>
1884 FUNCTION {output.eprint.url}
1885 {
1886   format.url output
1887 }
1888 </!*eprint>
1889

```

## 2.6 Entry types

Text below in this section is borrowed from biblatex manual. Not every field listed below is actually supported by GOST styles. So description below should be considered as a goal or a feature request.

The lists below indicate the fields supported by each entry type. Note that the mapping of fields to an entry type is ultimately at the discretion of the bibliography style. The lists below therefore serve two purposes. They indicate the fields supported by the standard styles which ship with this package and they also serve as a model for custom styles. Note that the required fields



are not strictly required in all cases. The fields marked as optional are optional in a technical sense. Bibliographical formatting rules usually require more than just the required fields. The standard styles will generally not perform any formal validity checks, but custom styles may do so. Generic fields like abstract and annotation or label and shorthand are not included in the lists below because they are independent of the entry type.

### 2.6.1 Regular Types

**article** An article in a journal, magazine, newspaper, or other periodical which forms a self-contained unit with its own title. The title of the periodical is given in the `journaltitle` field. If the issue has its own title in addition to the main title of the periodical, it goes in the `issuetitle` field. Note that `editor` and related fields refer to the journal while `translator` and related fields refer to the article.

Required fields: `author`, `title`, `journaltitle`, `year/date`.

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

```
1890 FUNCTION {article}
1891 {
1892   output.bibitem
1893   author.before
1894   (natbib) author format.key output
1895   format.title add.medium "title" output.check
1896   new.slash
1897   author.after
1898   new.dblslash
1899   journal emphasize add.doi "journal" output.check % new in v.2
1900   new.block
1901   format.date "year" output.check
1902   new.block
1903   format.volume output
1904   format.number output
1905   new.block
1906   format.pages.page output
1907   new.block
1908   note output
1909   new.sentence
1910 %   format.url output
1911   output.eprint.url
1912   format.annotate output
1913   fin.entry
1914 }
1915
```

**book** A single-volume book with one or more authors where the authors share credit for the work as a whole. In biblatex, this entry type also covers the function of the `@inbook` type of traditional BibTeX.

Required fields: `author`, `title`, `year/date`.

Optional fields: `editor`, `editora`, `editorb`, `editorc`, `translator`, `annotator`, `commentator`, `introduction`,

foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```

1916 FUNCTION {book}
1917 {
1918   output.bibitem
1919   author.before
1920 (natbib)  author format.key output
1921   format.btitle add.doi add.medium "title" output.check
1922   new.slash
1923   author.after
1924   editor.organization.after
1925   new.sentence
1926   format.number.series output
1927   new.block
1928   format.edition output
1929   new.block
1930   output.address.publisher
1931   format.date "year" output.check
1932   new.block
1933   format.bvolume output
1934   new.block
1935   format.pages.page output
1936   new.block
1937 (eprint)  format.isbn output
1938 (eprint)  new.block
1939   note output
1940   new.sentence
1941 %   format.url output
1942   output.eprint.url
1943   format.annotate output
1944   fin.entry
1945 }
1946

```

**booklet** A book-like work without a formal publisher or sponsoring institution. Use the field `howpublished` to supply publishing information in free format, if applicable. The field `type` may be useful as well.

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

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

```

1947 FUNCTION {booklet}
1948 {
1949   output.bibitem
1950   author.before
1951 (natbib)  author format.key output
1952   format.title add.doi add.medium "title" output.check
1953   new.slash
1954   author.after
1955   editor.organization.after
1956   new.block

```

```

1957 howpublished output
1958 address output
1959 format.date "year" output.check
1960 new.block
1961 note output
1962 new.sentence
1963 % format.url output
1964 output.eprint.url
1965 format.annotate output
1966 fin.entry
1967 }
1968

```

**inbook** A part of a book which forms a self-contained unit with its own title. Note that the profile of this entry type is different from standard BibTeX.

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

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

```

1969 FUNCTION {inbook}
1970 {
1971   output.bibitem
1972   author.before
1973   (natbib) author format.key output
1974   format.btitle add.doi add.medium "title" output.check
1975   new.slash
1976   author.after
1977   new.dblslash
1978   % bookauthor.before
1979   booktitle "booktitle" output.check
1980   new.slash
1981   bookauthor.after
1982   editor.organization.after
1983   new.block
1984   format.edition output
1985   new.block
1986   format.number.series output
1987   new.sentence
1988   output.address.publisher
1989   format.date "year" output.check
1990   new.block
1991   format.bvolume output
1992   new.block
1993   format.pages.page output
1994   new.block
1995   (eprint) format.isbn output
1996   (eprint) new.block
1997   note output

```

```

1998 new.sentence
1999 % format.url output
2000 output.eprint.url
2001 format.annotate output
2002 fin.entry
2003 }
2004

```

**incollection** A contribution to a collection which forms a self-contained unit with a distinct author and title. The author refers to the title, the editor to the booktitle, i. e., the title of the collection.

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

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

```

2005 FUNCTION {incollection}
2006 {
2007   output.bibitem
2008   author.before
2009 (natbib) author format.key output
2010 new.sentence
2011 format.title add.doi add.medium "title" output.check
2012 new.slash
2013 author.after
2014 new.dblslash
2015 booktitle "booktitle" output.check
2016 new.slash
2017 editor.organization.after
2018 new.block
2019 output.address.publisher
2020 format.date "year" output.check
2021 new.block
2022 format.bvolume output
2023 format.number.series output
2024 new.block
2025 format.pages.page output
2026 new.block
2027 note output
2028 new.sentence
2029 % format.url output
2030 output.eprint.url
2031 format.annotate output
2032 fin.entry
2033 }
2034

```

**proceedings** A single-volume conference proceedings. This type is very similar to @collection. It supports an optional organization field which holds the sponsoring institution. The editor is omissible.

Required fields: editor, title, year/date.

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

```

2035 FUNCTION {proceedings}
2036 {
2037   output.bibitem
2038   (natbib) editor format.key output
2039   format.btitle add.doi add.medium "title" output.check
2040   new.slash
2041   editor.organization.after
2042   new.block
2043   format.bvolume output
2044   format.number.series output
2045   % address empty$
2046   %   { publisher output
2047   %     format.date "year" output.check
2048   %   }
2049   %   { address output.nonnull
2050   %     format.date "year" output.check
2051   %     new.sentence
2052   %     publisher output
2053   %   }
2054   % if$
2055   output.address.publisher
2056   format.date "year" output.check
2057   new.block
2058   note output
2059   new.sentence
2060   % format.url output
2061   output.eprint.url
2062   format.annotate output
2063   fin.entry
2064 }
2065

```

**inproceedings** An article in a conference proceedings. This type is similar to @incollection. It supports an optional organization field.

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

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

```

2066 FUNCTION {inproceedings}
2067 { output.bibitem
2068   author.before
2069   (natbib) author format.key output
2070   new.sentence
2071   format.title add.doi add.medium "title" output.check
2072   new.slash

```

```

2073 author.after
2074 new.dblslash
2075 booktitle "booktitle" output.check
2076 new.slash
2077 editor.organization.after
2078 new.block
2079 format.bvolume output
2080 format.number.series output
2081 new.block
2082 % address empty$
2083 % { publisher output
2084 %   format.date "year" output.check
2085 % }
2086 % { address output.nonnull
2087 %   new.column
2088 %   publisher output
2089 %   format.date "year" output.check
2090 % }
2091 % if$
2092 output.address.publisher
2093 format.date "year" output.check
2094 new.block
2095 format.pages.page output
2096 new.block
2097 note output
2098 new.sentence
2099 % format.url output
2100 output.eprint.url
2101 format.annotate output
2102 fin.entry
2103 }
2104

```

**manual** Technical or other documentation, not necessarily in printed form. The author or editor is omissible.

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

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

```

2105 FUNCTION {manual}
2106 { output.bibitem
2107   author empty$
2108     { organization empty$
2109       'skip$
2110       { organization output.nonnull
2111         address output
2112       }
2113     } if$
2114   }
2115   { format.authors output.nonnull }
2116   if$

```

```

2117 <natbib> author format.key output
2118 new.block
2119 format.btitle add.doi add.medium "title" output.check
2120 author empty$
2121 { organization empty$
2122 {
2123     address new.block.checka
2124     address output
2125 }
2126 'skip$
2127 if$
2128 }
2129 {
2130     organization address new.block.checkb
2131     organization output
2132     address output
2133 }
2134 if$
2135 format.edition output
2136 format.date "year" output.check
2137 new.block
2138 note output
2139 new.sentence
2140 % format.url output
2141 output.eprint.url
2142 format.annotate output
2143 fin.entry
2144 }
2145

```

**misc** A fallback type for entries which do not fit into any other category. Use the field `howpublished` to supply publishing information in free format, if applicable. The field type may be useful as well. `author`, `editor`, and `year` are omissible.

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

```

2146 FUNCTION {misc}
2147 { output.bibitem
2148   format.authors output
2149   <natbib> author format.key output
2150   title howpublished new.sentence.checkb
2151   format.title add.medium output
2152   howpublished new.block.checka
2153   howpublished output
2154   new.block
2155   format.date "year" output.check
2156   new.block
2157   note output
2158   new.sentence
2159   % format.url output
2160   output.eprint.url
2161   format.annotate output

```

```

2162  fin.entry
2163 }
2164

```

**unpublished** A work with an author and a title which has not been formally published, such as a manuscript or the script of a talk. Use the fields `howpublished` and `note` to supply additional information in free format, if applicable.

Required fields: author, title, year/date.

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

```

2165 FUNCTION {unpublished}
2166 { output.bibitem
2167   author.before
2168    author format.key output
2169   format.btitle "title" output.check
2170   new.slash
2171   author.after
2172   editor.organization.after
2173   new.block
2174   format.date "year" output.check
2175   new.block
2176   note "note" output.check
2177   new.sentence
2178 %   format.url output
2179   output.eprint.url
2180   format.annote output
2181   fin.entry
2182 }
2183

```

**online** An online resource. Author, editor, and year are omissible. This entry type is intended for sources such as web sites which are intrinsically online resources. Note that all entry types support the `url` field. For example, when adding an article from an online journal, it may be preferable to use the `@article` type and its `url` field.

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

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

```

2184 FUNCTION {online}
2185 { output.bibitem
2186   format.authors output
2187    author format.key output
2188   title howpublished new.sentence.checkb
2189   format.title add.doi add.medium "title" output.check
2190 %   howpublished new.block.checka
2191   howpublished new.dblslash.checka
2192   howpublished enclose.square.brackets output
2193   editor.organization.after
2194   new.sentence
2195   new.block
2196   output.address.publisher

```



```

2197   format.date output
2198   new.block
2199 %   format.url output
2200   output.eprint.url
2201   new.sentence
2202   note output
2203   format.annotate output
2204   fin.entry
2205 }
2206

internet      New in version 2012.02.15.
  www 2207 FUNCTION {internet}   {online}
  webpage 2208 FUNCTION {www}     {online}
  ielectronic 2209 FUNCTION {webpage} {online}
  2210 FUNCTION {electronic} {online}

thesis      New in version 2012.02.02.
  A thesis written for an educational institution to satisfy the requirements for a degree. Use the
  type field to specify the type of thesis.
  Required fields: author, title, type, institution, year/date.
  Optional fields: subtitle, titleaddon, language, note, location, month, isbn, chapter, pages,
  pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

2211 <*(modern | strict)>
2212 FUNCTION {thesis}
2213 { output.bibitem
2214   format.authors "author" output.check
2215 <natbib> author format.key output
2216   new.sentence
2217   format.btitle "title" output.check
2218   new.column
2219   bbl.phdthesis format.thesis.type output.nonnull
2220   new.slash
2221   school "school" output.check
2222   new.block
2223   output.address.publisher.date
2224   new.block
2225   format.pages.page output
2226   note output
2227   new.sentence
2228   format.url output
2229   format.annotate output
2230   fin.entry
2231 }
2232 </!(modern | strict)>
2233 <*(modern | strict)>
2234 FUNCTION {thesis}
2235 { output.bibitem
2236   format.authors "author" output.check
2237 <natbib> author format.key output
2238   new.sentence

```

```

2239 format.btitle add.doi add.medium "title" output.check
2240 new.column
2241 % bbl.phdthesis format.thesis.type output.nonnull
2242 type "type" output.check
2243 new.column
2244 number output
2245 new.slash
2246 format.authors.after output
2247 new.semicolumn
2248 school "school" output.check
2249 new.block
2250 output.address.publisher
2251 format.date "year" output.check
2252 new.block
2253 format.pages.page output
2254 new.block
2255 note output
2256 new.sentence
2257 % format.url output
2258 output.eprint.url
2259 format.annotate output
2260 fin.entry
2261 }
2262 </modern | strict>
2263
2264 % \DescribeFunction{report}
2265 % New in version 2012.02.02.
2266 %
2267 % A technical report, research report, or white paper published by a university or
2268 % some other institution. Use the type field to specify the type of report. The sponsoring
2269 % institution goes in the institution field.
2270 %
2271 % Required fields: author, title, type, institution, year/date.
2272 %
2273 % Optional fields: subtitle, titleaddon, language, number, version, note,
2274 % location, month, isrn, chapter, pages, pagetotal, addendum, pubstate, doi,
2275 % eprint, eprintclass, eprinttype, url, urldate.
2276 % \begin{macrocode}
2277 %FUNCTION {report}
2278 %{
2279 % output.bibitem
2280 % author.before
2281 % new.sentence
2282 % format.title add.doi add.medium "title" output.check
2283 % new.column
2284 %% format.techrep.type.number output.nonnull
2285 % type "type" output.check
2286 % new.slash
2287 % author.after
2288 % editor.organization.after

```

```

2289 % new.block
2290 % address output
2291 % new.column
2292 % institution "institution" output.check
2293 % format.date "year" output.check
2294 % new.block
2295 % note output
2296 % new.block % v.2
2297 % format.pages.page output % v.2
2298 % new.sentence
2299 %% format.url output
2300 % output.eprint.url
2301 % format.annotate output
2302 % fin.entry
2303 %}
2304 FUNCTION {report}
2305 {
2306   output.bibitem
2307 %   author.before
2308 %   new.sentence
2309   format.title add.doi add.medium "title" output.check
2310 (natbib) title format.key output
2311   new.column
2312 %   format.techrep.type.number output.nonnull
2313 %   type "type" output.check
2314 %   format.report.type.number "type" output.check
2315   type "type" output.check
2316   new.column
2317   number output
2318   new.slash
2319   institution "institution" output.check
2320   new.semicolumn
2321   format.chief.after output % from editor field
2322   new.semicolumn
2323   format.executor.after output % from author field
2324   new.block
2325   address output
2326   new.column
2327   organization output
2328   format.date "year" output.check
2329   new.block % v.2
2330   format.pages.page output % v.2
2331   new.sentence % или new.block ?
2332   output.eprint.url
2333   new.block
2334   note output
2335   format.annotate output
2336   fin.entry
2337 }
2338

```

## 2.6.2 Type Aliases

The entry types listed in this section are provided for backwards compatibility with traditional BibTeX styles. These aliases are resolved by BibTeX as the data is exported. Bibliography styles will see the entry type the alias points to, not the alias name. All unknown entry types are generally exported as @misc.

**phdthesis** Similar to @thesis except that the type field is optional and defaults to the localized term ‘PhD thesis’. You may still use the type field to override that.

```

2339 <*(modern|strict)>
2340 FUNCTION {phdthesis}
2341 { output.bibitem
2342   format.authors "author" output.check
2343   new.sentence
2344   format.btitle "title" output.check
2345   new.column
2346   bbl.phdthesis format.thesis.type output.nonnull
2347   new.slash
2348   school "school" output.check
2349   new.block
2350   output.address.publisher.date
2351   new.block
2352   format.pages.page output
2353   note output
2354   new.sentence
2355   format.url output
2356   format.annotate output
2357   fin.entry
2358 }
2359 </!(modern|strict)>
2360 <*(modern|strict)>
2361 FUNCTION {phdthesis}
2362 { output.bibitem
2363   format.authors "author" output.check
2364   <natbib> author format.key output
2365   new.sentence
2366   format.btitle add.doi add.medium "title" output.check
2367   new.column
2368   bbl.phdthesis format.thesis.type output.nonnull
2369   new.column
2370   number output
2371   new.slash
2372   format.authors.after output
2373   new.semicolumn
2374   school "school" output.check
2375   new.block
2376   output.address.publisher
2377   format.date "year" output.check
2378   new.block
2379   format.pages.page output
2380   new.block

```

```

2381 note output
2382 new.sentence
2383 % format.url output
2384 output.eprint.url
2385 format.annotate output
2386 fin.entry
2387 }
2388 </modern | strict>
2389

```

**mastersthesis** Similar to @thesis except that the type field is optional and defaults to the localized term ‘Master’s thesis’. You may still use the type field to override that.

```

2390 <*(modern | strict)>
2391 FUNCTION {mastersthesis}
2392 { output.bibitem
2393   format.authors "author" output.check
2394 <natbib> author format.key output
2395   new.sentence
2396   format.btitle "title" output.check
2397   new.column
2398   bbl.mthesis format.thesis.type output.nonnull
2399   new.slash
2400   school "school" output.check
2401   new.block
2402   output.address.publisher.date
2403   new.block
2404   format.pages.page output
2405   note output
2406   new.sentence
2407   format.url output
2408   format.annotate output
2409   fin.entry
2410 }
2411 </!(modern | strict)>
2412 <*(modern | strict)>
2413 FUNCTION {mastersthesis}
2414 { output.bibitem
2415   format.authors "author" output.check
2416 <natbib> author format.key output
2417   new.sentence
2418   format.btitle add.doi add.medium "title" output.check
2419   new.column
2420   bbl.mthesis format.thesis.type output.nonnull
2421   new.column
2422   number output
2423   new.slash
2424   format.authors.after output
2425   new.semicolumn
2426   school "school" output.check
2427   new.block
2428   output.address.publisher

```

```

2429 format.date "year" output.check
2430 new.block
2431 format.pages.page output
2432 new.block
2433 note output
2434 new.sentence
2435 % format.url output
2436 output.eprint.url
2437 format.annotate output
2438 fin.entry
2439 }
2440 </modern | strict>
2441

```

**dscithesis** Similar to @thesis except that the type field is optional and defaults to the localized term ‘Doctor’s of sciences thesis’. You may still use the type field to override that.

```

2442 <*(modern | strict)>
2443 FUNCTION {dscithesis}
2444 { output.bibitem
2445   format.authors "author" output.check
2446 <natbib> author format.key output
2447   new.sentence
2448   format.btitle "title" output.check
2449   new.column
2450   bbl.dscithesis format.thesis.type output.nonnull
2451   new.slash
2452   school "school" output.check
2453   new.block
2454   output.address.publisher.date
2455   new.block
2456   format.pages.page output
2457   note output
2458   new.sentence
2459   format.url output
2460   format.annotate output
2461   fin.entry
2462 }
2463 </!(modern | strict)>
2464 <*modern | strict>
2465 FUNCTION {dscithesis}
2466 { output.bibitem
2467   format.authors "author" output.check
2468 <natbib> author format.key output
2469   new.sentence
2470   format.btitle add.doi add.medium "title" output.check
2471   new.column
2472   bbl.dscithesis format.thesis.type output.nonnull
2473   new.column
2474   number output
2475   new.slash
2476   format.authors.after output

```

```

2477 new.semicolumn
2478 school "school" output.check
2479 new.block
2480 output.address.publisher
2481 format.date "year" output.check
2482 new.block
2483 format.pages.page output
2484 new.block
2485 note output
2486 new.sentence
2487 % format.url output
2488 output.eprint.url
2489 format.annotate output
2490 fin.entry
2491 }
2492 </modern | strict>
2493

```

conference

```

2494 FUNCTION {conference} { inproceedings }
2495

```

**techreport**      TechReport is similar to @report except that the **type** field is optional and defaults to the localized term ‘technical report’. You may still use the **type** field to override that.

```

2496 %FUNCTION {techreport}
2497 %{
2498 % output.bibitem
2499 % author.before
2500 % new.sentence
2501 % format.title add.doi add.medium "title" output.check
2502 % new.column
2503 % format.techrep.type.number output.nonnull
2504 % new.slash
2505 % author.after
2506 % editor.organization.after
2507 % new.block
2508 % address output
2509 % new.column
2510 % institution "institution" output.check
2511 % format.date "year" output.check
2512 % new.block
2513 % note output
2514 % new.block % v.2
2515 % format.pages.page output % v.2
2516 % new.sentence
2517 %% format.url output
2518 % output.eprint.url
2519 % format.annotate output
2520 % fin.entry
2521 %}
2522

```

```

2523 FUNCTION {techreport}
2524 {
2525   output.bibitem
2526 %   author.before
2527 %   new.sentence
2528   format.title add.doi add.medium "title" output.check
2529 (natbib)   title format.key output
2530   new.column
2531 %   format.techrep.type.number output.nonnull
2532 %   type "type" output.check
2533 %   format.report.type.number "type" output.check
2534 %   type output
2535   format.techreport.type output
2536   new.column
2537   number output
2538   new.slash
2539   institution "institution" output.check
2540   new.semicolumn
2541   format.chief.after output % from editor field
2542   new.semicolumn
2543   format.executor.after output % from author field
2544   new.block
2545   address output
2546   new.column
2547   organization output
2548   format.date "year" output.check
2549   new.block % v.2
2550   format.pages.page output % v.2
2551   new.sentence % или new.block ?
2552   output.eprint.url
2553   new.block
2554   note output
2555   format.annotate output
2556   fin.entry
2557 }
2558
2559

```

default.type

```

2560 FUNCTION {default.type} { misc }
2561

```

## 2.7 Month Abbreviations

```

2562 MACRO {jan} {"\bbljan{}}
2563 MACRO {feb} {"\bblfeb{}}
2564 MACRO {mar} {"\bblmar{}}
2565 MACRO {apr} {"\bblapr{}}
2566 MACRO {may} {"\bblmay{}}
2567 MACRO {jun} {"\bbljun{}}
2568 MACRO {jul} {"\bbljul{}}

```



2569 MACRO {aug} {"\bblaug{}}"  
 2570 MACRO {sep} {"\bblsep{}}"  
 2571 MACRO {oct} {"\bbloct{}}"  
 2572 MACRO {nov} {"\bblnov{}}"  
 2573 MACRO {dec} {"\bbldec{}}"  
 2574

## 2.8 Journal Abbreviations

2575

## 2.9 Journal abbreviations

### 2.9.1 Physics and astronomy

Borrowed from physjour.mbs of package custom-bib.

2576 MACRO {aa} {"Astron. \& Astrophys."}  
 2577 MACRO {aasup} {"Astron. \& Astrophys. Suppl. Ser."}  
 2578 MACRO {aj} {"Astron. J."}  
 2579 MACRO {aph} {"Acta Phys."}  
 2580 MACRO {advp} {"Adv. Phys."}  
 2581 MACRO {ajp} {"Amer. J. Phys."}  
 2582 MACRO {ajm} {"Amer. J. Math."}  
 2583 MACRO {amsci} {"Amer. Sci."}  
 2584 MACRO {anofd} {"Ann. Fluid Dyn."}  
 2585 MACRO {am} {"Ann. Math."}  
 2586 MACRO {ap} {"Ann. Phys. (NY)"}

2587 MACRO {adp} {"Ann. Phys. (Leipzig)"}

2588 MACRO {ao} {"Appl. Opt."}

2589 MACRO {apl} {"Appl. Phys. Lett."}

2590 MACRO {app} {"Astroparticle Phys."}

2591 MACRO {apj} {"Astrophys. J."}

2592 MACRO {apjsup} {"Astrophys. J. Suppl."}

2593 MACRO {apss} {"Astrophys. Space Sci."}

2594 MACRO {araa} {"Ann. Rev. Astron. Astrophys."}

2595 MACRO {baas} {"Bull. Amer. Astron. Soc."}

2596 MACRO {baps} {"Bull. Amer. Phys. Soc."}

2597 MACRO {cmp} {"Comm. Math. Phys."}

2598 MACRO {cpam} {"Commun. Pure Appl. Math."}

2599 MACRO {cppcf} {"Comm. Plasma Phys. \& Controlled Fusion"}

2600 MACRO {cpc} {"Comp. Phys. Comm."}

2601 MACRO {cqg} {"Class. Quant. Grav."}

2602 MACRO {cra} {"C. R. Acad. Sci. A"}

2603 MACRO {fed} {"Fusion Eng. \& Design"}

2604 MACRO {ft} {"Fusion Tech."}

2605 MACRO {grg} {"Gen. Relativ. Gravit."}

2606 MACRO {ieeens} {"IEEE Trans. Nucl. Sci."}

2607 MACRO {ieeeps} {"IEEE Trans. Plasma Sci."}

2608 MACRO {ijimw} {"Interntl. J. Infrared \& Millimeter Waves"}

2609 MACRO {ip} {"Infrared Phys."}

2610 MACRO {irp} {"Infrared Phys."}

2611 MACRO {jap} {"J. Appl. Phys."}

2612 MACRO {jasa} {"J. Acoust. Soc. America"}  
2613 MACRO {jcp} {"J. Comp. Phys."}  
2614 MACRO {jchp} {"J. Chem. Phys."}  
2615 MACRO {jetp} {"Sov. Phys.--JETP"}  
2616 MACRO {jfe} {"J. Fusion Energy"}  
2617 MACRO {jfm} {"J. Fluid Mech."}  
2618 MACRO {jmp} {"J. Math. Phys."}  
2619 MACRO {jne} {"J. Nucl. Energy"}  
2620 MACRO {jnec} {"J. Nucl. Energy, C: Plasma Phys., Accelerators, Thermonucl. Res."}  
2621 MACRO {jnm} {"J. Nucl. Mat."}  
2622 MACRO {jpc} {"J. Phys. Chem."}  
2623 MACRO {jpp} {"J. Plasma Phys."}  
2624 MACRO {jpsj} {"J. Phys. Soc. Japan"}  
2625 MACRO {jsi} {"J. Sci. Instrum."}  
2626 MACRO {jvst} {"J. Vac. Sci. & Tech."}  
2627 MACRO {nat} {"Nature"}  
2628 MACRO {nature} {"Nature"}  
2629 MACRO {nedf} {"Nucl. Eng. \& Design/Fusion"}  
2630 MACRO {nf} {"Nucl. Fusion"}  
2631 MACRO {nim} {"Nucl. Inst. \& Meth."}  
2632 MACRO {nimp} {"Nucl. Inst. \& Meth. in Phys. Res."}  
2633 MACRO {np} {"Nucl. Phys."}  
2634 MACRO {npb} {"Nucl. Phys. B"}  
2635 MACRO {nt/f} {"Nucl. Tech./Fusion"}  
2636 MACRO {npbpc} {"Nucl. Phys. B (Proc. Suppl.)"}  
2637 MACRO {inc} {"Nuovo Cimento"}  
2638 MACRO {nc} {"Nuovo Cimento"}  
2639 MACRO {pf} {"Phys. Fluids"}  
2640 MACRO {pfa} {"Phys. Fluids A: Fluid Dyn."}  
2641 MACRO {pfb} {"Phys. Fluids B: Plasma Phys."}  
2642 MACRO {pl} {"Phys. Lett."}  
2643 MACRO {pla} {"Phys. Lett. A"}  
2644 MACRO {plb} {"Phys. Lett. B"}  
2645 MACRO {prep} {"Phys. Rep."}  
2646 MACRO {pnas} {"Proc. Nat. Acad. Sci. USA"}  
2647 MACRO {pp} {"Phys. Plasmas"}  
2648 MACRO {ppcf} {"Plasma Phys. \& Controlled Fusion"}  
2649 MACRO {phitrs1} {"Philos. Trans. Roy. Soc. London"}  
2650 MACRO {pr1} {"Phys. Rev. Lett."}  
2651 MACRO {pr} {"Phys. Rev."}  
2652 MACRO {physrev} {"Phys. Rev."}  
2653 MACRO {pra} {"Phys. Rev. A"}  
2654 MACRO {prb} {"Phys. Rev. B"}  
2655 MACRO {prc} {"Phys. Rev. C"}  
2656 MACRO {prd} {"Phys. Rev. D"}  
2657 MACRO {pre} {"Phys. Rev. E"}  
2658 MACRO {ps} {"Phys. Scripta"}  
2659 MACRO {procrs1} {"Proc. Roy. Soc. London"}  
2660 MACRO {rmp} {"Rev. Mod. Phys."}  
2661 MACRO {rsi} {"Rev. Sci. Inst."}

```

2662 MACRO {science} {"Science"}
2663 MACRO {sciam} {"Sci. Am."}
2664 MACRO {sam} {"Stud. Appl. Math."}
2665 MACRO {sjpp} {"Sov. J. Plasma Phys."}
2666 MACRO {spd} {"Sov. Phys.--Doklady"}
2667 MACRO {sptp} {"Sov. Phys.--Tech. Phys."}
2668 MACRO {spu} {"Sov. Phys.--Uspekhi"}
2669 MACRO {st} {"Sky and Telesc."}
2670

```

## 2.9.2 Optics

Borrowed from photjour.mbs.

```

2671 MACRO {appopt} {"Appl. Opt."}
2672 MACRO {bell} {"Bell Syst. Tech. J."}
2673 MACRO {ell} {"Electron. Lett."}
2674 MACRO {jasp} {"J. Appl. Spectr."}
2675 MACRO {jqe} {"IEEE J. Quantum Electron."}
2676 MACRO {jlwt} {"J. Lightwave Technol."}
2677 MACRO {jmo} {"J. Mod. Opt."}
2678 MACRO {josa} {"J. Opt. Soc. America"}
2679 MACRO {josaa} {"J. Opt. Soc. Amer.~A"}
2680 MACRO {josab} {"J. Opt. Soc. Amer.~B"}
2681 MACRO {jdp} {"J. Phys. (Paris)"}
2682 MACRO {oc} {"Opt. Commun."}
2683 MACRO {ol} {"Opt. Lett."}
2684 MACRO {os} {"Opt. Spectrosc."}
2685 MACRO {phtl} {"IEEE Photon. Technol. Lett."}
2686 MACRO {pspie} {"Proc. Soc. Photo-Opt. Instrum. Eng."}
2687 MACRO {sjot} {"Sov. J. Opt. Technol."}
2688 MACRO {sjqe} {"Sov. J. Quantum Electron."}
2689 MACRO {sleb} {"Sov. Phys.--Leb. Inst. Rep."}
2690 MACRO {stph} {"Sov. Phys.--Techn. Phys."}
2691 MACRO {stphl} {"Sov. Techn. Phys. Lett."}
2692 MACRO {vr} {"Vision Res."}
2693 MACRO {zph} {"Z. f. Physik"}
2694 MACRO {zphb} {"Z. f. Physik~B"}
2695 MACRO {zphd} {"Z. f. Physik~D"}

```

## 2.9.3 Physics of condensed Matter

```

2696 MACRO {sse} {"Solid-State Electron."}
2697 MACRO {pss} {"Phys. Sol. State"}
2698 MACRO {sst} {"Semicond. Sci. Tech."}
2699 MACRO {nl} {"Nano Lett."}
2700
2701 READ
2702

```

## 2.10 Sorting

Next chunk of code governs sorting reference list by authors' names and titles.

```
2703 ⟨*sort | natbib⟩
2704
```

sortify

```
2705 FUNCTION {sortify}
2706 { purify$
2707 ⟨!utf8⟩ "1" change.case$
2708 }
2709 ⟨/sort | natbib⟩
2710
```

sort.format.names

```
2711 ⟨*sort⟩
2712 %% =====
2713 %% This version from old Gost package
2714 %%<!*natbib>
2715 FUNCTION {sort.format.names}
2716 { 's :=
2717   #1 'nameptr :=
2718   ""
2719   s num.names$ 'numnames :=
2720   numnames 'namesleft :=
2721   { namesleft #0 > }
2722   { nameptr #1 >
2723     { " " * }
2724     'skip$
2725     if$
2726     s nameptr
2727     "{vv{ } }{ll{ }}{ f{ }}{ jj{ }}"
2728     format.name$ 't :=
2729     nameptr numnames = t "others" = and
2730     { "et al" * }
2731     %{ bbl.etal * }
2732     { t sortify * }
2733     if$
2734     nameptr #1 + 'nameptr :=
2735     namesleft #1 - 'namesleft :=
2736   }
2737   while$
2738 }
2739 %%</!natbib>
2740 %% This version from plainnat.bst
2741 %% It ignores second and subsequent authors but include year.
2742 %%<!*natbib>
2743 %FUNCTION {sort.format.names}
2744 %{ 's :=
2745 %   #1 'nameptr :=
2746 %   ""
```

```

2747 % s num.names$ 'numnames :=
2748 % numnames 'namesleft :=
2749 % { namesleft #0 > }
2750 % {
2751 %     s nameptr "{vv{ } }{ll{ }}{ ff{ }}{ jj{ }}" format.name$ 't :=
2752 %     nameptr #1 >
2753 %     {
2754 %         " " *
2755 %         namesleft #1 = t "others" = and
2756 %         { "zzzzz" * }
2757 %         { numnames #2 > nameptr #2 = and
2758 %             { "zz" * year field.or.null * " " * }
2759 %             'skip$
2760 %             if$
2761 %             t sortify *
2762 %         }
2763 %     if$
2764 %     }
2765 %     { t sortify * }
2766 %     if$
2767 %     nameptr #1 + 'nameptr :=
2768 %     namesleft #1 - 'namesleft :=
2769 %     }
2770 % while$
2771 %}
2772 %%</natbib>
2773 %% =====
2774

```

sort.format.title

```

2775 FUNCTION {sort.format.title}
2776 { 't :=
2777   "A " #2
2778   "An " #3
2779   "The " #4 t chop.word % Removes "The " if any
2780   chop.word % Removes "An " if any
2781   chop.word % Removes "A " if any
2782   sortify
2783   #1 global.max$ substring$
2784 }
2785

```

author.sort

```

2786 %% =====
2787 %% This version from old gost package.
2788 %%
2789 <!*natbib>
2790 FUNCTION {author.sort}
2791 { author empty$
2792   { key empty$
2793     { "to sort, need author or key in " cite$ * warning$

```

```

2794         ""
2795     }
2796     { key sortify }
2797     if$
2798 }
2799 {
2800     author num.names$ #4 <
2801         {author sort.format.names }
2802         {title sort.format.title}
2803     if$
2804 }
2805 if$
2806 }
2807 <\/!natbib>
2808 %% This version from plainnat.bst
2809 <natbib>
2810 FUNCTION {author.sort}
2811 { author empty$
2812     { key empty$
2813         { "to sort, need author or key in " cite$ * warning$
2814             ""
2815         }
2816         { key sortify }
2817     if$
2818     }
2819     { author sort.format.names }
2820 if$
2821 }
2822 <\/natbib>
2823 %% =====
2824

```

author.title.sort

```

2825 <natbib | natbib>
2826 FUNCTION {author.title.sort}
2827 { author empty$
2828     { title empty$
2829         { key empty$
2830             { "to sort, need author, title, or key in " cite$ * warning$
2831                 ""
2832             }
2833             { key sortify }
2834         if$
2835         }
2836         { title sort.format.title }
2837     if$
2838     }
2839     {
2840         author num.names$ #4 <
2841             {author sort.format.names }
2842             {title sort.format.title}

```

```

2843     if$
2844 }
2845 if$
2846 }
2847 </!natbib|natbib>
2848 <*natbib|natbib>
2849 FUNCTION {author.editor.sort}
2850 { author empty$
2851   { editor empty$
2852     { key empty$
2853       { "to sort, need author, editor, or key in " cite$ * warning$
2854         ""
2855       }
2856       { key sortify }
2857     }
2858   }
2859   { editor sort.format.names }
2860 } if$
2861 }
2862 { author sort.format.names }
2863 if$
2864 }
2865
2866 FUNCTION {author.organization.sort}
2867 { author empty$
2868   { organization empty$
2869     { key empty$
2870       { "to sort, need author, organization, or key in " cite$ * warning$
2871         ""
2872       }
2873       { key sortify }
2874     }
2875   }
2876   { "The " #4 organization chop.word sortify }
2877 } if$
2878 }
2879 { author sort.format.names }
2880 if$
2881 }
2882
2883 FUNCTION {editor.organization.sort}
2884 { editor empty$
2885   { organization empty$
2886     { key empty$
2887       { "to sort, need editor, organization, or key in " cite$ * warning$
2888         ""
2889       }
2890       { key sortify }
2891     }
2892   }

```

```

2893     { "The " #4 organization chop.word sortify }
2894     if$
2895   }
2896   { editor sort.format.names }
2897   if$
2898 }
2899 </natbib | natbib>
2900
2901

```

presort    Function to compute sort.key\$. What is the space string "\_\_\_" for?

```

2902 <!*natbib>
2903 FUNCTION {presort}%#1
2904 {
2905   author.title.sort
2906   "   "
2907   *
2908   year field.or.null sortify
2909   *
2910   "   "
2911   *
2912   title field.or.null
2913   sort.format.title
2914   *
2915   #1 entry.max$ substring$
2916   'sort.key$ :=
2917 }
2918 </!natbib>
2919 <!*natbib>
2920 FUNCTION {presort}%#2
2921 { calc.label
2922   label sortify
2923   %author.title.sort
2924   "   "
2925   *
2926   % ===== plainnat.bst =====
2927   % type$ "book" =
2928   % type$ "inbook" =
2929   % or
2930   %   'author.editor.sort
2931   %   { type$ "proceedings" =
2932   %     'editor.organization.sort
2933   %     { type$ "manual" =
2934   %       'author.organization.sort
2935   %       'author.sort
2936   %     if$
2937   %   }
2938   %   if$
2939   % }
2940   % if$
2941   author.title.sort

```



```

2942 "      "
2943 *
2944 year field.or.null sortify
2945 *
2946 "      "
2947 *
2948 %cite$
2949 title field.or.null sort.format.title
2950 *
2951 #1 entry.max$ substring$
2952 'sort.label :=
2953 sort.label *
2954 % =====
2955 #1 entry.max$ substring$
2956 'sort.key$ :=
2957 }
2958 </natbib>
2959 </sort>
2960
2961 <!*sort>
2962 <!*natbib>
2963 INTEGERS { seq.num }
2964
2965 FUNCTION {init.seq}
2966 { #0 'seq.num :=}
2967
2968 EXECUTE {init.seq}
2969
2970 FUNCTION {int.to.fix}
2971 { "000000000" swap$ int.to.str$ *
2972  #-1 #10 substring$
2973 }
2974
2975 FUNCTION {presort}%#3
2976 {
2977   calc.label           % computes label
2978   label sortify        % initiates sort.label
2979   "      "
2980   *
2981   seq.num #1 + 'seq.num := % advance seq.num
2982   seq.num int.to.fix      % prepend seq.num with 0s
2983   'sort.label :=         % set sort.label to seq.num
2984   sort.label *           % append seq.num to label
2985   #1 entry.max$ substring$ % cut if too long
2986   'sort.key$ :=         % set sort.key$
2987 }
2988 </natbib>
2989 </!sort>
2990
2991 <*sort | natbib>

```

```

2992 ITERATE {presort}
2993
2994 SORT
2995
2996 </sort | natbib>
2997

```

## 2.11 Bibliography list

We need to find longest label to put in into the argument of the `thebibliography` environment. In case of `natbib` options we also need to compute extra suffix for the `year` field if there two or more entries for given label (=author/editor/organization) in that year.

Declare global (external) strings used in calculation of the longest label.

```

2998 <!natbib>STRINGS { longest.label }
2999 <natbib>STRINGS { longest.label last.label next.extra }
3000
3001 <!natbib>INTEGERS { number.label longest.label.width }
3002 <natbib>INTEGERS { number.label longest.label.width last.extra.num }
3003

```

`initialize.longest.label` Initialize those string.

```

3004 <!*natbib>
3005 FUNCTION {initialize.longest.label}
3006 { "" 'longest.label :=
3007   #1 'number.label :=
3008   #0 'longest.label.width :=
3009 }
3010 </!natbib>
3011 <*natbib>
3012 FUNCTION {initialize.longest.label}
3013 { "" 'longest.label :=
3014   #0 int.to.chr$ 'last.label :=
3015   "" 'next.extra :=
3016   #0 'longest.label.width :=
3017   #0 'last.extra.num :=
3018   #0 'number.label :=
3019 }
3020 </natbib>
3021
3022 EXECUTE {initialize.longest.label}
3023

```

`initialize.longest.label` Iterate though the list of entries to compute label.

```

3024 <!*natbib>
3025 FUNCTION {forward.pass}
3026 { number.label int.to.str$ 'label :=
3027   number.label #1 + 'number.label :=
3028   label width$ longest.label.width >
3029   { label 'longest.label :=
3030     label width$ 'longest.label.width :=

```

```

3031     }
3032     'skip$
3033     if$
3034 }
3035 </!natbib>
3036 <*natbib>
3037 FUNCTION {forward.pass}
3038 { last.label label =
3039   { last.extra.num #1 + 'last.extra.num :=
3040     last.extra.num int.to.chr$ 'extra.label :=
3041   }
3042   { "a" chr.to.int$ 'last.extra.num :=
3043     "" 'extra.label :=
3044     label 'last.label :=
3045   }
3046   if$
3047   number.label #1 + 'number.label :=
3048 }
3049 </natbib>
3050
3051 ITERATE {forward.pass}
3052

```

reverse.pass      Natbib styles require reverse iteration over all entries.

```

3053 <*natbib>
3054 FUNCTION {reverse.pass}
3055 { next.extra "b" =
3056   { "a" 'extra.label := }
3057   'skip$
3058   if$
3059   extra.label 'next.extra :=
3060   extra.label
3061   duplicate$ empty$
3062   'skip$
3063   { "{\natexlab{" swap$ * "}}" * }
3064   if$
3065   'extra.label :=
3066   label extra.label * 'label :=
3067 }
3068
3069 REVERSE {reverse.pass}
3070
3071 FUNCTION {bib.sort.order}
3072 { sort.label 'sort.key$ :=
3073 }
3074
3075 ITERATE {bib.sort.order}
3076
3077 SORT
3078 </natbib>
3079

```

`begin.bib` Within the `thebibliography` environment we define few formatting macros for user to customize how the reference list is formatted.

```

3080 FUNCTION {begin.bib}
3081 { "\begin{thebibliography}{\longest.label *}" * write$ newline$
3082   "\def\selectlanguageifdefined#1{" write$ newline$
3083   "\expandafter\ifx\csname date#1\endcsname\relax" write$ newline$
3084   "\else\language\csname l@#1\endcsname\fi" write$ newline$
3085   "\providecommand*\href{[2]{\small #2}}" write$ newline$
3086   "\providecommand*\url{[1]{\small #1}}" write$ newline$
3087   "\providecommand*\BibUrl{[1]{\url{#1}}}" write$ newline$
3088   "\providecommand*\BibAnnote{[1]{}}" write$ newline$
3089   "\providecommand*\BibEmph{[1]{#1}}" write$ newline$
3090 <*modern>
3091   "\providecommand*\cyrdash{\hbox to.8em{--\hss--}}" write$ newline$
3092   "\providecommand*\BibDash{\ifdim\lastskip>Opt\unskip\nobreak\hskip.2em\fi} write$ newline$
3093   "\cyrdash\hskip.2em\ignorespaces" write$ newline$
3094 </modern>
3095 <natbib> "\providecommand*\natexlab{[1]{#1}} write$ newline$
3096   preamble$ empty$
3097   'skip$
3098   { preamble$ write$ newline$ }
3099   if$
3100 }
3101
3102
3103 EXECUTE {begin.bib}
3104
3105 EXECUTE {init.state.consts}
3106
3107 ITERATE {call.type$}
3108

```

`end.bib`

```

3109 FUNCTION {end.bib}
3110 { newline$
3111 % "\catcode'\/=11" write$ newline$
3112 "\end{thebibliography}" write$ newline$
3113 }
3114
3115 EXECUTE {end.bib}
3116
3117 </bst>

```

That's all, Folks!

## Change History

v0.8		\BibEmph added . . . . . 1
General: \BibAnnote added . . . . . 1		\BibUrl added . . . . . 1

Entry ANNOTE added	1	v1.2	
v0.9			General: Entries eprint, eprintclass, eprinttype
General: Bug fix in INPROCEEDINGS	1		1
Bug fix in names and date formatting	1		1
v1.0			Fix bbl.urldate for ukrainian (Andrey Shvajkoy)
General: Bug fix (long annote)	1		1
v1.1			Medium field
General: Added German, French, Italian languages	1		1
Entry ONLINE	1		Options modern, long, eprint
Gost705.dtx borrowed from Disser pkg	1		1
Upload to CTAN	1		Refactoring, Documentation
			Strict option
			Support for natbib package
			1
			Thesis entry, report entry
			1

### 3 Index

Numbers written in dark blue refer to the page where the corresponding entry is described; numbers in black roman refer to the code lines where the entry is used.

Symbols			
\&	2576, 2577, 2599, 2603, 2608, 2626, 2629, 2631, 2632, 2648	bbl.medium.elres	23
\'	400	bbl.medium.text	23
\,	696, 698, 705, 713, 720	bbl.medium	24
\/	3111	bbl.mthesis	21
add.blank	11	bbl.nnoaddress	22
add.doi	46	bbl.nnopublisher	22
add.medium	46	bbl.nnr	18
and	11	bbl.nnumber	17
article	49	bbl.nopublisher	22
author.after	43	bbl.nr	18
author.before	42	bbl.number	18
author.editor.full	33	bbl.of	16
author.full	33	bbl.pages	19
author.key.label	27	bbl.page	19
author.sort	69	bbl.phdthesis	21
author.title.sort	70	bbl.ppages	20
bbl.and	17	bbl.ppage	20
bbl.chief	23	bbl.techreport	21
bbl.cmplr	14	bbl.urldate	20
bbl.dscithesis	22	bbl.vvolume	15
bbl.edby	14	bbl.vvol	15
bbl.edition	15	begin.bib	76
bbl.etal	17	bookauthor.after	44
bbl.executor	24	bookauthor.before	43
bbl.iin	19	booklet	50
bbl.iissue	16	book	49
bbl.iiss	16	change.language	8
bbl.in	19	chop.word	11
		conference	63
		convert.edition	38
		default.type	64
		dscithesis	62
		editor.full	33
		editor.organization.after	44
		either.or.chec	36
		emphasize	13
		empty.misc.check	41
		enclose.square.brackets	13
		end.bib	76
		eng.ord	38
		extract.num	37
		field.or.null	12
		fin.entry	9
		format.address.publisher.date	31
		format.address.publisher	31
		format.annote	45
		format.authors.after	29
		format.authors	27
		format.bookauthors.after	29
		format.bookauthors	29
		format.btitle	36
		format.bvolume	36
		format.chapter.pages	41
		format.chief.after	30
		format.compiler.after	30
		format.date	30
		format.edition	39
		format.editors.after	29
		format.eprint	47
		format.executor.after	30
		format.full.names	33



<code>\hskip</code> . . . . .	3092, 3093	<b>N</b>	<b>S</b>
<code>\hss</code> . . . . .	3091	<code>\natexlab</code> . . . . .	3063, 3095
<b>I</b>		<code>\newblock</code> . . . . .	117
<code>\ifdim</code> . . . . .	3092	<code>\nobreak</code> . . . . .	3092
<code>\ifx</code> . . . . .	3083	<b>P</b>	
<code>\ignorespaces</code> . . . . .	3093	<code>\providecommand</code> . . .	3085,
<b>L</b>		3086, 3087, 3088,	<b>T</b>
<code>\language</code> . . . . .	3084	3089, 3091, 3092, 3095	<code>\textnumero</code> . . . . .
<code>\lastskip</code> . . . . .	3092	<b>R</b>	554, 568
<code>\ldots</code> . . . . .	700, 702, 717, 719	<code>\relax</code> . . . . .	3083
		<b>U</b>	
		<code>\unskip</code> . . . . .	3092
		<code>\url</code> . . . . .	3086, 3087