

# The GOST package

Igor A. Kotelnikov\*

2016/07/25, v.1.2g

## 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Kotelnikov to the present status and some code was borrowed from `disser` package developed by Stanislav Kruchinin and unpublished work by Artem Petrenkov.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. 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 `bibtex.exe` nor `bibtex8.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 1.2g (2016.07.25)

1. Minor changes in documentation.

### 1.5 What's new in version 1.2f (2016.07.12)

1. Support for patent entry added (thanks to Stanislav Kruchinin).
2. medium field renamed to media field.

### 1.6 What's new in version 1.2e (2016.07.07)

1. Hard coded "URL" string replaced with a language sensitive string (thanks to Roman Budnyi).

### 1.7 What's new in version 1.2d (2015.02.18)

1. jan, feb, e.t.c. macros fixed.
2. New macro format.month.

### 1.8 What's new in version 1.2c (2015.01.10)

1. `langid` field is added. It has same meaning as `language` which is now obsolete but is still supported for backward compatibility; `langid` has priority over `language`.
2. `eid` field is added. It has priority over `pages`.
3. The ligature `---` has been substituted with `\BibDash` for `.bst` styles compiled without `modern` options (`gost2003.bst` and `gost2003s.bst`). For modern styles this was done in earlier versions.
4. Spacing around `\BibDash` has been improved.
5. `\BibDash` now typesets short em-dash (`\cyrdahs`) only for `ruussian` and `ukrainian` languages. In earlier versions, it produces short em-dash for all languages.  
(This feature was removed since it did not work with all engines.)

## 1.9 What's new in version 1.2a (2012.08.31)

1. `\cyrdash` is now defined via `\ProvideTextCommand` rather than `\providecommand`.

## 1.10 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. All modern styles are now compiled with the `eprint` option.

## 1.11 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.

## 1.12 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 %% \cite{key} => (Jones et al., 1990)
25 %% \cite*{key} => (Jones, Baker, and Smith, 1990)
26 %% \cite[chap. 2]{key} => (Jones et al., 1990, chap. 2)
27 %% \cite[e.g.][key] => (e.g. Jones et al., 1990)
28 %% \cite[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 \end{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   eid % new in v1.2c

```

```

55 publisher
56 school
57 series
58 title
59 %medium % new in v1.2; renamed to media.
60 media % new in v1.2f
61 type
62 volume
63 year
64 language
65 langid % new in v1.2c
66 booklanguage

```

Entries borrowed from biblatex.

```

67 date % new in v1.2f; not implemented yet...
68 pagetotal
69 url
70 urldate
71 isbn
72 doi
73 % archive
74 eprinttype % = archivePrefix
75 eprintclass % = primaryClass
76 eprint

```

Entries borrowed from disser.bst by S.Kruchinin.

```

77 % new in v1.2f:
78 % appear in biblatex:
79 addendum
80 holder
81 location
82 subtitle
83 titleaddon
84 version
85 % Appear in biblatex-gost:
86 authorcountry % ??
87 credits % statement of responsibility, other than provided in Biblatex
88 ipc % Code of the International Patent Classification
89 %media % General material designation NOTE: medium in the above
90 requestnumber % Registration number of the application to the patent document
91 publicationdate % Date of publication
92 publication % and information on the official gazette, which published patent
93 prioritydate % Information about the convention priority: the date of filing of the application,
94 prioritynumber % number and
95 prioritycountry % country name of convention priority.
96 requestdate % ??
97 }
98 {}
99 <natbib> { label }
100 <natbib> { label extra.label sort.label short.list }
101

```



Declare internal variables and constants used in to format references.

```
102 INTEGERS { output.state before.all mid.sentence after.sentence after.block
103 after.dblslash after.slash after.colon after.semicolon }
104
```

init.state.consts

```
105 FUNCTION {init.state.consts}
106 { #0 'before.all :=
107   #1 'mid.sentence :=
108   #2 'after.sentence :=
109   #3 'after.block :=
110   #4 'after.dblslash :=
111   #5 'after.slash :=
112   #6 'after.colon :=
113   #7 'after.semicolon :=
114 }
115
116 STRINGS { s t }
117
118 STRINGS { curlanguage }
119
```

## 2.2 Formatting functions

change.language    Declare function to switch language.

```
120 FUNCTION {change.language}
121 { booklanguage empty$
122   { "" }
123   { booklanguage 'curlanguage :=
124     "\selectlanguageifdefined{"
125     curlanguage *
126     "}" *
127   }
128   if$
129 }
130
```

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

```
131 FUNCTION {output.nonnull}
132 {
133   swap$
134   output.state mid.sentence =
135   { ", " * write$ }
136   { output.state after.block =
137     { add.period$ write$
138       " \BibDash " write$
139       newline$
140       "\newblock " write$
141     }
142     { output.state before.all =
```

```

143         'write$
144         { output.state after.dblslash =
145           { "~/ " * change.language * " " * write$ }
146         { output.state after.slash =
147           { "~/ " * write$ }
148         { output.state after.colon =
149           { ": " * write$ }
150         { "~: " * write$ }
151         { output.state after.semicolon =
152           { "; " * write$ }
153         { "~; " * write$ }
154         { add.period$ " " * write$ }
155         if$
156         }
157         if$
158         }
159         if$
160         }
161         if$
162         }
163         if$
164         }
165         if$
166         mid.sentence 'output.state :=
167         }
168         if$
169     }
170
output
output.check 171 FUNCTION {output}
172 { duplicate$ empty$
173   'pop$
174   'output.nonnull
175   if$
176 }
177
178 FUNCTION {output.check}
179 { 't :=
180   duplicate$ empty$
181   { pop$ "empty " t * " in " * cite$ * warning$ }
182   'output.nonnull
183   if$
184 }
185
fin.entry    fin.entry finalizes current entry. It writes dot, if no dot is found in stack, and starts new line.
186 FUNCTION {fin.entry}
187 { add.period$
188   write$
189   newline$

```

```

190 }
191
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.
192 FUNCTION {new.block}
193 { output.state before.all =
194   'skip$
195   { after.block 'output.state := }
196   if$
197 }
198
new.dblslash
199 FUNCTION {new.dblslash}
200 { output.state before.all =
201   'skip$
202   { after.dblslash 'output.state := }
203   if$
204 }
205
new.slash
206 FUNCTION {new.slash}
207 { output.state before.all =
208   'skip$
209   { after.slash 'output.state := }
210   if$
211 }
212
new.colon
213 FUNCTION {new.colon}
214 { output.state before.all =
215   'skip$
216   { after.colon 'output.state := }
217   if$
218 }
219
new.semicolon
220 FUNCTION {new.semicolon}
221 { output.state before.all =
222   'skip$
223   { after.semicolon 'output.state := }
224   if$
225 }
226
new.sentence
227 FUNCTION {new.sentence}

```

```

228 { output.state after.block =
229     'skip$
230     { output.state before.all =
231         'skip$
232         { after.sentence 'output.state := }
233         if$
234     }
235     if$
236 }
237

```

add.blank

```

238 FUNCTION {add.blank}
239 { " " * before.all 'output.state :=
240 }
241

```

not        Declare few logical functions.

```

242 FUNCTION {not}
243 { { #0 }
244   { #1 }
245   if$
246 }
247

```

and

```

248 FUNCTION {and}
249 { 'skip$
250   { pop$ #0 }
251   if$
252 }
253

```

or

```

254 FUNCTION {or}
255 { { pop$ #1 }
256   'skip$
257   if$
258 }
259

```

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.

```

260 (*sort | natbib)
261 INTEGERS { len }
262
263 FUNCTION {chop.word}
264 { 's :=
265   'len :=
266   s #1 len substring$ =
267     { s len #1 + global.max$ substring$ }
268     's

```

```

269   if$
270 }
271 </sort | natbib>
272
non.stop
273 FUNCTION {non.stop}
274 { duplicate$
275   "}" * add.period$
276   #-1 #1 substring$ "." =
277 }
278
new.block.checka
279 FUNCTION {new.block.checka}
280 { empty$
281   'skip$
282   'new.block
283   if$
284 }
285
new.block.checkb
286 FUNCTION {new.block.checkb}
287 { empty$
288   swap$ empty$
289   and
290   'skip$
291   'new.block
292   if$
293 }
294
w.sentence.checka
295 FUNCTION {new.sentence.checka}
296 { empty$
297   'skip$
298   'new.sentence
299   if$
300 }
301
w.sentence.checkb
302 FUNCTION {new.sentence.checkb}
303 { empty$
304   swap$ empty$
305   and
306   'skip$
307   'new.sentence
308   if$
309 }
310

```

```

new.dblslash.checka    For online entry.
311 FUNCTION {new.dblslash.checka}
312 { empty$
313   'skip$
314   'new.dblslash
315   if$
316 }
317

field.or.null

318 FUNCTION {field.or.null}
319 { duplicate$ empty$
320   { pop$ "" }
321   'skip$
322   if$
323 }
324

emphasize              Declare function to emphasize last string in stack.
325 FUNCTION {emphasize}
326 { duplicate$ empty$
327   { pop$ "" }
328   { "\BibEmph{" swap$ * "}" * }
329   if$
330 }
331

bracify                New in v.1.2. An idea borrowed from apsrev4-1.bst. Declare function to enclose last word by
braces even if empty.
332 FUNCTION {bracify}
333 { duplicate$ empty$
334   { pop$ "{}" }
335   { "{" swap$ * "}" * }
336   if$
337 }
338

e.square.brackets      This and the next functions are used to enclose last word by brackets.
339 FUNCTION {enclose.square.brackets}
340 {
341   duplicate$ empty$
342   { pop$ "" }
343   { "[" swap$ * "]" * }
344   if$
345 }
346

e.round.brackets

347 FUNCTION {enclose.round.brackets}
348 {
349   duplicate$ empty$
350   { pop$ "" }

```

```

351 { "(" swap$ * ")" * }
352 if$
353 }
354

```

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

```

355 FUNCTION {space.word}
356 { " " swap$ * " " * }
357

```

## 2.3 Standard abbreviations

bb1.edby    Declare language-sensitive abbreviations.

```

358 FUNCTION {bb1.edby}    % { "\bbledby{}" }
359 { curlanguage "english" =
360   {"Ed.\ by"}
361   { curlanguage "ukrainian" =
362     {!\utf8}        {"{\cyr\CYRP\cyrii\cyrd\ \cyrr\cyre\cyrd.}" }
363     {!\utf8}        {"Під ред." }
364     { curlanguage "russian" =
365       {!\utf8}        {"{\cyr\CYRP\cyro\cyrd\ \cyrr\cyre\cyrd.}" }
366       {!\utf8}        {"Под ред." }
367       { curlanguage "german" =
368         { "ed." }
369         {"language is not defined: " language "edby" * * warning$ "Ed.\ by"}
370         if$}
371       if$}
372     if$}
373 if$}
374

```

bb1.cmplr

```

375 FUNCTION {bb1.cmplr}
376 { curlanguage "english" =
377   { "Compiler" }
378   { curlanguage "german" =
379     { "Hrsg." }
380     { curlanguage "ukrainian" =
381       {!\utf8}        {"{\cyr\CYRU\cyrk\cyrl.}" }
382       {!\utf8}        {"{Укл.}" }
383       { curlanguage "russian" =
384         {!\utf8}        {"{\cyr\CYRS\cyro\cyrs\cyrt.}" }
385         {!\utf8}        {"{Сост.}" }
386         {"language is not defined: " language "cmplr" * * warning$ "Compiler"}
387         if$}
388       if$}
389     if$}
390 if$}
391

```

bb1.edition

```

392 FUNCTION {bbledition} % { "\bbledition{" }
393 { curlanguage "english" =
394   {"edition"}
395   { curlanguage "ukrainian" =
396     <utf8> {"{\cyr\cyrv\cyri\cyrd.}" }
397     <utf8> {"{\вйд.}" }
398     { curlanguage "russian" =
399       <utf8> {"{\cyr\cyri\cyrz\cyrd.}" }
400       <utf8> {"{\изд.}" }
401       { curlanguage "german" =
402         {"{aus.}" } %%% { "Auf1." } ??
403         { curlanguage "italian" =
404           {"edizione"}
405           { curlanguage "french" =
406             {"\'{e}dition"}
407             {"language is not defined: " language "edition" * * warning$ "edition"}
408             if$}
409             if$}
410             if$}
411             if$}
412             if$}
413 if$}
414

```

bb1.vvolume

```

415 FUNCTION {bb1.vvolume} % { "\bb1Volume{" }
416 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
417   {"Volume"}
418   { curlanguage "ukrainian" = curlanguage "russian" = or
419     <utf8> {"{\CYRT\cyro\cyrm" }
420     <utf8> {"{Tom" }
421     { curlanguage "german" =
422       {"{Band}" } %%% { "Volumen" }
423       {"language is not defined: " language "vvolume" * * warning$ "Volume"}
424       if$}
425       if$}
426 if$}
427

```

bb1.vvol

```

428 FUNCTION {bb1.vvol} % { "\bb1Vol{" }
429 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
430   {"Vol."}
431   { curlanguage "ukrainian" = curlanguage "russian" = or
432     <utf8> {"{\CYRT."}
433     <utf8> {"{T."}
434     { curlanguage "german" =
435       {"{Bd.}" } %%% { "Vol." }
436       {"language is not defined: " language "vvol" * * warning$ "Vol."}
437       if$}
438       if$}

```



```

439 if$}
440

```

bbl.iissue

```

441 FUNCTION {bbl.iissue} % { "\bblIssue{" }
442 { curlanguage "english" =
443   {"Issue"}
444   { curlanguage "ukrainian" =
445     <utf8> {"\CYRV\cyri\cyrp\cyru\cyrs\cyrk"}
446     <utf8> {"Випуск"}
447     { curlanguage "russian" =
448       <utf8> {"\CYRV\cyrrery\cyrp\cyru\cyrs\cyrk"}
449       <utf8> {"Выпуск"}
450       { curlanguage "german" =
451         {"{Heft}"} %%% { "Ausgabe" }
452         {"language is not defined: " language "iissue" * * warning$ "Issue"}
453         if$}
454       if$}
455     if$}
456 if$}
457

```

bbl.iiss

```

458 FUNCTION {bbl.iiss} % { "\bblIss{" }
459 { curlanguage "english" =
460   {"Iss."}
461   { curlanguage "ukrainian" =
462     <utf8> {"\CYRV\cyri\cyrp."}
463     <utf8> {"Вип."}
464     { curlanguage "russian" =
465       <utf8> {"\CYRV\cyrrery\cyrp."}
466       <utf8> {"Вип."}
467       { curlanguage "german" =
468         {"{H.}"}
469         {"language is not defined: " language "iiss" * * warning$ "Iss."}
470         if$}
471       if$}
472     if$}
473 if$}
474

```

bbl.of

```

475 FUNCTION {bbl.of} % { "\bblOf{" }
476 { curlanguage "english" =
477   {"of"}
478   { curlanguage "german" =
479     {"von" }
480     { curlanguage "ukrainian" =
481       <utf8> {"{\cyr\cyrii\cyrz}" }
482       <utf8> {"{із}" }
483       { curlanguage "russian" =
484         <utf8> {"{\cyr\cyri\cyrz}" }

```

```

485 <utf8>          { "{\u0430}" }
486          {"language is not defined: " language "of" * * warning$ "of"}
487          if$}
488          if$}
489          if$}
490 if$}
491

```

bbl.etal

```

492 FUNCTION {bbl.etal} % { "\bblof{}" }
493 { curlanguage "english" =
494   {"et~al."}
495   { curlanguage "german" =
496     { "u.~a." }
497     { curlanguage "ukrainian" =
498       <!utf8>      {"{\cyr\cyrt\cyra~\cyrii\cyrn.}" }
499       <utf8>      {"{\cyr\cyri~\cyrd\cyrr.}" }
500       { curlanguage "russian" =
501         <!utf8>    {"{\cyr\cyri~\cyrd\cyrr.}" }
502         <utf8>    {"{\cyr\cyri~\cyrd\cyrr.}" }
503         {"language is not defined: " language "et~al" * * warning$ "et~al."}
504         if$}
505         if$}
506         if$}
507 if$}
508

```

bbl.and

```

509 <natbib>
510 FUNCTION {bbl.and} % { "\bbland{}" }
511 { curlanguage "english" =
512   {"and"}
513   { curlanguage "german" =
514     { "und" }
515     { curlanguage "ukrainian" =
516       <!utf8>      {"{\cyrii}" }
517       <utf8>      {"i"}
518       { curlanguage "russian" =
519         <!utf8>    {"{\cyri}" }
520         <utf8>    {"\u0430"}
521         {"language is not defined: " language "and" * * warning$ "and"}
522         if$}
523         if$}
524         if$}
525 if$}
526 </natbib>
527

```

bbl.number

```

528 FUNCTION {bbl.number} % { "\bblNumber{}" }
529 { curlanguage "english" =
530   {"Number"}

```

```

531 { curlanguage "ukrainian" = curlanguage "russian" = or
532 <!utf8> { "\CYRN\cyro\cyrm\cyre\cyrr" }
533 <utf8> { "{Homep}" }
534 { curlanguage "german" =
535     {"{Heft}"} %%% { "Anzahl" }
536     {"language is not defined: " language "nnumber" * * warning$ "Number"}
537     if$}
538     if$}
539 if$}
540

```

bbl.number

```

541 FUNCTION {bbl.number} % { "\bblnumber{" }
542 { curlanguage "english" =
543     {"number"}
544     { curlanguage "ukrainian" = curlanguage "russian" = or
545 <!utf8> { "{\cyr\cyrn\cyro\cyrm\cyre\cyrr}" }
546 <utf8> { "{номер}" }
547     { curlanguage "german" =
548         {"{Heft}"} %%% { "anzahl" } ???
549         {"language is not defined: " language "number" * * warning$ "number"}
550         if$}
551         if$}
552 if$}
553

```

bbl.nr

```

554 FUNCTION {bbl.nr} % { "\bblno{" }
555 { curlanguage "english" =
556     {"no."}
557     { curlanguage "italian" =
558         {"no" }
559         { curlanguage "ukrainian" = curlanguage "russian" = or
560 <!utf8> { "{\cyr\textnumero}" }
561 <utf8> { "{№}" }
562         { curlanguage "german" =
563             {"{H.}"} %%% { "an." }
564             {"language is not defined: " language "nr" * * warning$ "no."}
565             if$}
566             if$}
567         if$}
568 if$}
569

```

bbl.nnr

```

570 FUNCTION {bbl.nnr} % { "\bblno{" }
571 { curlanguage "english" =
572     {"No."}
573     { curlanguage "ukrainian" = curlanguage "russian" = or
574 <!utf8> { "{\cyr\textnumero}" }
575 <utf8> { "{№}" }
576     { curlanguage "german" =

```

```

577         {"{H.}" } %%% { "an." }
578         {"language is not defined: " language "nnr" * * warning$ "No."}
579     if$}
580 if$}
581 if$}
582

```

bb1.in

```

583 FUNCTION {bb1.in} % { "\bb1in{" }
584 { curlanguage "english" = curlanguage "german" = or
585     {"in"}
586     { curlanguage "ukrainian" = curlanguage "russian" = or
587     <utf8> { "{\cyr\cyrv}" }
588     <utf8> { "{B}" }
589     {"language is not defined: " language "in" * * warning$ "in"}
590     if$}
591 if$}
592

```

bb1.iin

```

593 FUNCTION {bb1.iin} % { "\bb1in{" }
594 { curlanguage "english" = curlanguage "german" = or
595     {"In"}
596     { curlanguage "ukrainian" = curlanguage "russian" = or
597     <utf8> { "{\CYRV}" }
598     <utf8> { "{B}" }
599     {"language is not defined: " language "iin" * * warning$ "In"}
600     if$}
601 if$}
602

```

bb1.pages

```

603 FUNCTION {bb1.pages} % { "\bb1pp." }
604 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
605     {"p." } %%% {"pp."}
606     { curlanguage "ukrainian" = curlanguage "russian" = or
607     <utf8> { "{\cyr\cyrs.}" }
608     <utf8> { "{c.}" }
609     { curlanguage "german" =
610     {"S." } %%% { "s." }
611     {"language is not defined: " language "pages" * * warning$ "p."}
612     if$}
613     if$}
614 if$}
615

```

bb1.page

```

616 FUNCTION {bb1.page} % { "\bb1p." }
617 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
618     {"p." }
619     { curlanguage "ukrainian" = curlanguage "russian" = or
620     <utf8> { "{\cyr\cyrs.}" }

```

```

621 <utf8>      {"{c.}" }
622      { curlanguage "german" =
623        {"S."} %%%% { "s." }
624        {"language is not defined: " language "page" * * warning$ "p."}
625      if$}
626    if$}
627 if$}
628

```

bb1.ppages

```

629 FUNCTION {bb1.ppages}%      { "\bb1P." }
630 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
631   {"P."} %%%% { "Pp." }
632   { curlanguage "ukrainian" = curlanguage "russian" = or
633     <!utf8>      {"{\cyr\CYRS.}" }
634     <utf8>      {"{C.}" }
635     { curlanguage "german" =
636       {"S."}
637       {"language is not defined: " language "ppages" * * warning$ "P."}
638     if$}
639   if$}
640 if$}
641

```

bb1.ppage

```

642 FUNCTION {bb1.ppage} %      { "\bb1P." }
643 { curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
644   {"P."}
645   { curlanguage "ukrainian" = curlanguage "russian" = or
646     <!utf8>      {"{\cyr\CYRS.}" }
647     <utf8>      {"{C.}" }
648     { curlanguage "german" =
649       {"S."}
650       {"language is not defined: " language "ppage" * * warning$ "P."}
651     if$}
652   if$}
653 if$}
654

```

bb1.url Next function was added in version 2016.07.07.

```

655 FUNCTION {bb1.url}
656 { curlanguage "english" =
657   {"Access mode"}
658   { curlanguage "ukrainian" =
659     <!utf8>      { "{\CYRR\cyre\cyrrzh\cyri\cyrm\ \cyrd\cyro\cyrs\cyrt\cyru\cyrp\cyru}" }
660     <utf8>      { "{Режим доступу}" }
661     { curlanguage "russian" =
662       <!utf8>      { "{\CYRR\cyre\cyrrzh\cyri\cyrm\ \cyrd\cyro\cyrs\cyrt\cyru\cyrp\cyra}" }
663       <utf8>      { "{Режим доступа}" }
664       { curlanguage "german" =
665         {"{online; abgerufen}" }
666         {"language is not defined: " language "urldate" * * warning$ "online; accessed" }

```

```

667         if$}
668     if$}
669 if$}
670 if$}

```

bb1.urldate Next function was added in version 2012.01.15.

```

671 FUNCTION {bb1.urldate}
672 { curlanguage "english" =
673   {"online; accessed"}
674   { curlanguage "ukrainian" =
675     {\utf8} { "{\cyrd\cyra\cyrt\cyra\ \cyrz\cyrv\cyre\cyrr\cyrn\cyre\cyrn\cyrn\cyrya}" }
676     {\utf8} { "{дата звернення}" }
677     { curlanguage "russian" =
678       {\utf8} { "{\cyrd\cyra\cyrt\cyra\ \cyro\cyrb\cyrr\cyra\cyrshch\cyre\cyrn\cyri\cyrya}" }
679       {\utf8} { "{дата обращения}" }
680       { curlanguage "german" =
681         {"online; abgerufen"}
682         {"language is not defined: " language "urldate" * * warning$ "online; accessed" }
683         if$}
684       if$}
685     if$}
686   if$}
687

```

bb1.techreport

```

688 FUNCTION {bb1.techreport} % rename to bb1.techreport
689 { curlanguage "english" =
690   {"Rep." }
691   { curlanguage "german" =
692     {"Bericht" }
693     { curlanguage "russian" =
694       {\utf8} { "{\cyr\CYRO\cyrt\cyrch\cyre\cyrt}" }
695       {\utf8} { "{Отчет}" }
696       {"language is not defined: " language "techrep" * * warning$ "Rep." }
697       if$}
698     if$}
699   if$}
700

```

bb1.mthesis

```

701 FUNCTION {bb1.mthesis}
702 { curlanguage "english" =
703   {"Master's thesis" }
704   { curlanguage "german" =
705     {"Diss.~Mag." }
706     { curlanguage "russian" =
707       {\utf8} { "{\cyr\CYRK\cyrv\cyra\cyr\cyri\cyrf\cyri\cyrk\cyra\cyrc\cyri"
708       {\utf8} { "\cyro\cyrn\cyrn\cyra\cyrya\ \cyrr\cyra\cyrb\cyro\cyrt\cyra\ " *
709       {\utf8} { "\cyrm\cyra\cyrg\cyri\cyrs\cyrt\cyrr\cyra}" * }
710       {\utf8} { "{Квалификационная работа магистра}" }
711       {"language is not defined: " language "mthesis" * * warning$ "Master's thesis" }
712       if$}

```

```

713   if$}
714 if$}
715

```

bbl.phdthesis

```

716 FUNCTION {bbl.phdthesis}
717 { curlanguage "english" =
718   { "Ph.\,D. thesis" }
719   { curlanguage "german" =
720     { "Diss.\~Ph.\,D." }
721     { curlanguage "russian" =
722       { "\cyr\CYRD\cyri\cyrs\cyrs\ldots\ \cyrk\cyra\cyrn\cyrd\cyri"
723         "\cyrd\cyra\cyrt\cyra\ \cyrn\cyra\cyru\cyrk}" * }
724       { "\Дисс\ldots\ кандидата наук}" }
725       { curlanguage "french" =
726         { "Th\'{e}se de doctorat" }
727         { "language is not defined: " language "phdthesis" * * warning$ "Ph.\,D. thesis" }
728         if$}
729       if$}
730     if$}
731 if$}
732

```

bbl.dscithesis

```

733 FUNCTION {bbl.dscithesis}
734 { curlanguage "english" =
735   { "Dr.\,Sci. dissertation" }
736   { curlanguage "german" =
737     { "Diss.\~Dr." }
738     { curlanguage "russian" =
739       { "\cyr\CYRD\cyri\cyrs\cyrs\ldots\ \cyrd\cyro\cyrk\cyrt\cyro"
740         "\cyrr\cyra\ \cyrn\cyra\cyru\cyrk}" * }
741       { "\Дисс\ldots\ доктора наук}" }
742       { "language is not defined: " language "dscithesis" * * warning$ "Dr.\,Sci. dissertation" }
743       if$}
744     if$}
745 if$}
746

```

bbl.nnoaddress

```

747 FUNCTION {bbl.nnoaddress}
748 { curlanguage "english" =
749   { "S.\ 1." }
750   { curlanguage "russian" =
751     { "\cyr\CYRB.\ \cyrm.}" }
752     { "\Б.\ м." }
753     { "language is not defined: " language "nnoaddress" * * warning$ "S.\ 1." }
754     if$}
755 if$}
756

```

bbl.nopublisher

```

757 FUNCTION {bbl.nopublisher}
758 { curlanguage "english" =
759   { "s.\ n." }
760   { curlanguage "russian" =
761     { "\cyr\cyrb.\ \cyri." }
762     { "{б.\ н.}" }
763     { "language is not defined: " language "nopublisher" * * warning$ "s.\ n." }
764     if$}
765 if$}
766

```

bbl.nopublisher

```

767 FUNCTION {bbl.nnopublisher}
768 { curlanguage "english" =
769   { "S.\ n." }
770   { curlanguage "russian" =
771     { "\cyr\CYRB.\ \cyri." }
772     { "{Б.\ Н.}" }
773     { "language is not defined: " language "nnopublisher" * * warning$ "S.\ n." }
774     if$}
775 if$}
776

```

bbl.media.text

```

777 FUNCTION {bbl.media.text}
778 { curlanguage "english" =
779   { "Text" }
780   { curlanguage "russian" = curlanguage "ukrainian" = or
781     { "\cyr\CYRT\cyre\cyrk\cyrs\cyrt." }
782     { "{Текст}" }
783     { "language is not defined: " language "media" * * warning$ "Text" }
784     if$}
785 if$}
786

```

bbl.media.elres

```

787 FUNCTION {bbl.media.elres}
788 { curlanguage "english" =
789   { "Electronic resource" }
790   { curlanguage "russian" =
791     { "\cyr\CYREREV\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyrrery\cyrishrt\ \cyrr\cyre\cyrs\cyru\cyrr" }
792     { "{Электронный ресурс}" }
793     { curlanguage "ukrainian" =
794       { "\cyr\CYRE\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyri\cyrishrt\ \cyrr\cyre\cyrs\cyru\cyrr" }
795       { "{Електронний ресурс}" }
796       { "language is not defined: " language "media" * * warning$ "Electronic resource" }
797       if$}
798     if$}
799 if$}
800

```

bbl.chief



```

801 FUNCTION {bbl.chief}
802 { curlanguage "english" =
803   { "chief" }
804   { curlanguage "russian" =
805     <utf8> { "\cyrr\cyru\cyrk." }
806     <utf8> { "{пук.}" }
807     { curlanguage "ukrainian" =
808       <utf8> { "\cyrr\cyru\cyrk." }
809       <utf8> { "{пук.}" }
810       { "language is not defined: " language "chief" * * warning$ "chief" }
811       if$}
812     if$}
813 if$}
814

```

bbl.executor

```

815 FUNCTION {bbl.executor}
816 { curlanguage "english" =
817   { "Executor" }
818   { curlanguage "russian" =
819     <utf8> { "{\cyr\cyri\cyrs\cyrp\cyro\cyr1\cyrn.}" }
820     <utf8> { "{исполн.}" }
821     { curlanguage "ukrainian" =
822       <utf8> { "{\cyr\cyrv\cyri\cyrk\cyro\cyrn\cyra\cyrv\cyre\cyrc\cyrsftsн}" }
823       <utf8> { "{виконавець}" }
824       { "language is not defined: " language "executor" * * warning$ "executor" }
825       if$}
826     if$}
827 if$}
828

```

bbl.media

```

829 FUNCTION {bbl.media}
830 { type$ "online" =
831   { bbl.media.elres }
832   { bbl.media.text }
833 if$}
834

```

bbl.req

```

835 FUNCTION {bbl.req}
836 {
837   curlanguage "english" =
838     { "req." }
839     { curlanguage "german" =
840       { "ang." }
841       { curlanguage "russian" =
842         <utf8> { "{\cyr\cyrz\cyra\cyrya\cyrv\cyr1.}" }
843         <utf8> { "{заявл.}" }
844         { "language is not defined: " language "req" * * warning$ "req" }
845         if$
846       }
847     }
848   }
849 }

```

```

847     if$
848     }
849     if$
850 }
851

```

bb1.publ

```

852 FUNCTION {bb1.publ}
853 {
854   curlanguage "english" =
855   { "publ." }
856   { curlanguage "german" =
857     { "ausg." }
858     { curlanguage "russian" =
859       { "\cyr\cyro\cyrp\cyru\cyrb\cyrl." } }
860       { "\cyr\cyro\cyrp\cyru\cyrb\cyrl." } }
861     { "language is not defined: " language "publication" * * warning$ "publication" }
862     if$
863   }
864   if$
865   }
866   if$
867 }
868

```

bb1.priority

```

869 FUNCTION {bb1.priority}
870 {
871   curlanguage "english" =
872   { "priority" }
873   { curlanguage "german" =
874     { "Prioritat" }
875     { curlanguage "russian" =
876       { "\cyr\cyrp\cyrr\cyri\cyro\cyrr\cyri\cyrt\cyre\cyrt." } }
877       { "\cyr\cyrp\cyrr\cyri\cyro\cyrr\cyri\cyrt\cyre\cyrt." } }
878     { "language is not defined: " language "priority" * * warning$ "priority" }
879     if$
880   }
881   if$
882   }
883   if$
884 }
885

```

## 2.4 Formatting functions

Declare functions to format separate elements of bibliographic reference.

```

886 INTEGERS { nameptr namesleft numnames }
887
888

```

`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 format 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 unicoded 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.

```
889 <!*modern>
890 FUNCTION {format.names}
891 {
892 <!utf8> #1 "{vv~}{ll}{~jj}{,~f.}" format.name$
893 <utf8> #1 "{vv~}{ll}{~jj}{,~ff}" format.name$
894 }
895 </!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'.

```
896 <*modern>
897 FUNCTION {format.names}
898 {
899   's :=
900   #1 'nameptr :=
901   s num.names$ 'numnames :=
902   numnames 'namesleft :=
903   { namesleft #0 > }
904   { s nameptr
905 <!utf8> "{vv~}{ll}{~jj}{~f.}" format.name$ 't :=
906 <utf8> "{vv~}{ll}{~jj}{~ff}" format.name$ 't :=
907     nameptr #1 >
908     { nameptr #4 =
909       numnames #4 > and
910       { "others" 't :=
911         #1 'namesleft := }
912       'skip$
913       if$
914       namesleft #1 >
915       { ", " * t * }
916       { t "others" =
917         t "~others" =
918         or
919 <!strict> { " " * bbl.etal * }
920 <strict> { " " * bbl.etal enclose.square.brackets *}
921         { ", " * t * }
922         if$
923       }
924       if$
925     }
926     't
927     if$
```

```

928     nameptr #1 + 'nameptr :=
929     namesleft #1 - 'namesleft :=
930 }
931 while$
932 }
933 </modern>
934

```

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

```

935 <natbib>
936 FUNCTION {format.lab.names}
937 { 's :=
938   language empty$
939   { "english" 'curlanguage := }
940   { language 'curlanguage := }
941   if$
942   s #1 "{vv~}{ll}" format.name$
943   s num.names$ duplicate$
944   #2 >
945   %{ pop$ " et~al." * }
946   { pop$ " " bbl.etal * * }
947   { #2 <
948     'skip$
949     { s #2 "{ff }{vv }{ll}{ jj}" format.name$ "others" =
950       %{ " et~al." * }
951       { " " bbl.etal * * }
952       %{ " and " * s #2 "{vv~}{ll}" format.name$ * }
953       { " " bbl.and " " * * * s #2 "{vv~}{ll}" format.name$ * }
954     if$
955   }
956   if$
957 }
958 if$
959 }
960 </natbib>
961

```

**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.

```

962 FUNCTION {format.names.rev}
963 {
964   's :=
965   #1 'nameptr :=
966   s num.names$ 'numnames :=
967   numnames 'namesleft :=
968   { namesleft #0 > }
969   { s nameptr
970     <utf8>      %"{f.}{~vv}{~ll}{, jj}" format.name$ 't :=
971     <utf8>      "{ff}{~vv}{~ll}{, jj}" format.name$ 't :=

```

```

972 <utf8>      "{ff}{~vv}{~ll}{, jj}" format.name$ 't :=
973     nameptr #1 >
974     { nameptr #4 =
975       numnames #4 > and
976       { "others" 't :=
977         #1 'namesleft := }
978       'skip$
979     if$
980     namesleft #1 >
981     { ", " * t * }
982     { t "others" =
983     t "~others" =
984     or
985     (!strict)      { " " * bbl.etal * }
986     (strict)      { " " * bbl.etal enclose.square.brackets * }
987     { ", " * t * }
988     if$
989     }
990     if$
991     }
992     't
993     if$
994     nameptr #1 + 'nameptr :=
995     namesleft #1 - 'namesleft :=
996     }
997   while$
998 }
999

```

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

```

1000 <*natbib>
1001 FUNCTION {format.key}
1002 { empty$
1003   { key field.or.null }
1004   { "" }
1005   if$
1006 }
1007 </natbib>
1008

```

**format.authors**

```

1009 FUNCTION {format.authors}
1010 { author empty$
1011   { "" }
1012   { author format.names emphasize}
1013   if$
1014 }
1015

```

**author.key.label**

```

1016 <*natbib>
1017 FUNCTION {author.key.label}

```

```

1018 { author empty$
1019   { key empty$
1020     { cite$ #1 #3 substring$ }
1021     'key
1022     if$
1023   }
1024   { author format.lab.names }
1025   if$
1026 }
1027
1028 FUNCTION {author.editor.key.label}
1029 { author empty$
1030   { editor empty$
1031     { key empty$
1032       { cite$ #1 #3 substring$ }
1033       %'key          %% causes lost of year
1034       { "{}" key * } %% Bug in bibtex8 ??
1035       if$
1036     }
1037     { editor format.lab.names }
1038     if$
1039   }
1040   { author format.lab.names }
1041   if$
1042 }
1043
1044 FUNCTION {author.key.organization.label}
1045 { author empty$
1046   { key empty$
1047     { organization empty$
1048       { cite$ #1 #3 substring$ }
1049       { "The " #4 organization chop.word #3 text.prefix$ }
1050       if$
1051     }
1052     'key
1053     if$
1054   }
1055   { author format.lab.names }
1056   if$
1057 }
1058
1059 FUNCTION {editor.key.organization.label}
1060 { editor empty$
1061   { key empty$
1062     { organization empty$
1063       { cite$ #1 #3 substring$ }
1064       { "The " #4 organization chop.word #3 text.prefix$ }
1065       if$
1066     }
1067     'key

```

```

1068     if$
1069   }
1070   { editor format.lab.names }
1071   if$
1072 }
1073
1074 FUNCTION {calc.short.authors}
1075 { type$ "book" =
1076   type$ "inbook" =
1077   or
1078     'author.editor.key.label
1079   { type$ "proceedings" =
1080     'editor.key.organization.label
1081     { type$ "manual" =
1082       'author.key.organization.label
1083       'author.key.label
1084     } if$
1085   }
1086   if$
1087 }
1088 if$
1089 'short.list :=
1090 }
1091
1092 FUNCTION {calc.label}
1093 { calc.short.authors
1094   short.list
1095   "("
1096   *
1097   year duplicate$ empty$
1098   short.list key field.or.null = or
1099     { pop$ "" }
1100     'skip$
1101   if$
1102   *
1103   'label :=
1104 }
1105
1106 </natbib>
1107

```

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

```

1108 FUNCTION {format.bookauthors}
1109 { bookauthor empty$
1110   { "" }
1111   { bookauthor format.names}
1112   if$
1113 }
1114

```

mat.authors.after

```
1115 FUNCTION {format.authors.after}
1116 { author empty$
1117   { "" }
1118   { author format.names.rev}
1119   if$
1120 }
1121
```

bookauthors.after

```
1122 FUNCTION {format.bookauthors.after}
1123 { bookauthor empty$
1124   { "" }
1125   { bookauthor format.names.rev}% always cuts to 4 persons
1126   if$
1127 }
1128
```

mat.editors.after

```
1129 FUNCTION {format.editors.after}
1130 { editor empty$
1131   { "" }
1132   { bbl.edby "\ " * editor format.names.rev * }
1133   if$
1134 }
1135
```

ormat.chief.after

```
1136 FUNCTION {format.chief.after}
1137 { editor empty$
1138   { "" }
1139   { bbl.chief "\ " * editor format.names.rev * }
1140   if$
1141 }
1142
```

mat.executor.after

```
1143 FUNCTION {format.executor.after}
1144 { author empty$
1145   { "" }
1146   { bbl.executor ": " * author format.names.rev * }
1147   if$
1148 }
1149
```

mat.compiler.after

```
1150 FUNCTION {format.compiler.after}
1151 { compiler empty$
1152   { "" }
1153   { bbl.cmplr "\ " * compiler format.names.rev * }
1154   if$
1155 }
1156
```



`format.title`    **Important note** Neither `bibtex` nor `bibtex8` can handle unicoded 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.

```
1157 FUNCTION {format.title}
1158 { title empty$
1159   { "" }
1160 <utf8> { title "t" change.case$ }
1161 <utf8> { title }
1162 if$
1163 }
1164
```

`format.month`    New in version 1.2d. This macro reads month field and translate English names of months to Russian if current language is Russian.

```
1165 FUNCTION {format.month}
1166 { month empty$
1167   { "" }
1168   { curlanguage "russian" =
1169     { month "Jan." =
1170       <utf8> { "\CYRYA\cyrn\cyrv." }
1171       <utf8> { "ЯНВ." }
1172     { month "Feb." =
1173       <utf8> { "\CYRF\cyre\cyrv\cyrr." }
1174       <utf8> { "ФЕВ." }
1175     { month "Mar." =
1176       <utf8> { "\CYRM\cyra\cyrr\cyrt" }
1177       <utf8> { "МАР" }
1178     { month "Apr." =
1179       <utf8> { "\CYRA\cyrp\cyrr." }
1180       <utf8> { "АПР." }
1181     { month "May" =
1182       <utf8> { "\CYRM\cyra\cyrishrt" }
1183       <utf8> { "МАЙ" }
1184     { month "Jun." =
1185       <utf8> { "\CYRI\cyryu\cyrn\cyrstsn" }
1186       <utf8> { "ИЮНЬ" }
1187     { month "Jul." =
1188       <utf8> { "\CYRI\cyryu\cyrl\cyrstsn" }
1189       <utf8> { "ИЮЛЬ" }
1190     { month "Aug." =
1191       <utf8> { "\CYRA\cyrv\cyrg\." }
1192       <utf8> { "АВГ." }
1193     { month "Sep." =
1194       <utf8> { "\CYRS\cyre\cyn\cyrt." }
1195       <utf8> { "СЕНТ." }
1196     { month "Oct." =
1197       <utf8> { "\CYRO\cyrk\cyrt." }
1198       <utf8> { "ОКТ." }
1199     { month "Nov." =
1200 <utf8> { "\CYRN\cyro\cyrya\cyrb." }

```

[illegible]

```
format.date
```

```

1211 FUNCTION {format.date}
1212 { year empty$
1213   { month empty$
1214     { "" }
1215     { "there's a month but no year in " cite$ * warning$
1216       format.month
1217     }
1218   if$
1219 }
1220 { month empty$
1221   'year
1222   { year ". \BibDash " format.month * * }
1223   if$
1224 }
1225 if$
1226 (natbib) extra.label * % new in v.1.2
1227 }
1228

```

address.publisher

```

1229 <!*strict>
1230 FUNCTION {output.address.publisher}
1231 {
1232     address empty$
1233     'skip$
1234     { address output
1235         publisher empty$
1236         'skip$
1237         { new.colon }
1238         if$
1239     }
1240     if$
1241     publisher output
1242 }
1243 </!strict>
1244 <!*strict>
1245 FUNCTION {output.address.publisher}
1246 {
1247     address empty$

```

```

1248 {
1249     bbl.nnoaddress
1250     publisher empty$
1251     { "~: " * bbl.nopublisher * enclose.square.brackets }
1252     { enclose.square.brackets "~: " * publisher * }
1253     if$
1254 }
1255 {
1256     address output
1257     new.colon
1258     publisher empty$
1259     { bbl.nopublisher enclose.square.brackets }
1260     { publisher }
1261     if$
1262 }
1263 if$
1264 output
1265 }
1266
1267 </strict>
1268

```

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

```

1269 <*(modern | strict)>
1270 FUNCTION {output.address.publisher.date}
1271 {
1272     output.address.publisher
1273     format.date output
1274 }
1275 </!(modern | strict)>
1276

```

output.bibitem

```

1277 <*(natbib)>
1278 FUNCTION {output.bibitem}
1279 { newline$
1280     "\bibitem" write$
1281     cite$ bracing write$
1282     newline$
1283     langid empty$
1284     { language empty$
1285         { "english" 'curlanguage := }
1286         { language 'curlanguage := }
1287     if$
1288     }
1289     { langid 'curlanguage := }
1290     if$
1291     "\selectlanguageifdefined" curlanguage bracing * write$
1292     newline$
1293     ""
1294     before.all 'output.state :=

```

```

1295 }
1296
1297 </!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            1298 <natbib>
author.editor.full      1299 FUNCTION {format.full.names}
make.full.names        1300 { 's :=
output.bibitem         1301   language empty$
                        1302   { "english" 'curlanguage := }
                        1303   { language 'curlanguage := }
                        1304   if$
                        1305   #1 'nameptr :=
                        1306   s num.names$ 'numnames :=
                        1307   numnames 'namesleft :=
                        1308   { namesleft #0 > }
                        1309   { s nameptr
                        1310     "{vv~}{ll}" format.name$ 't :=
                        1311     nameptr #1 >
                        1312     {
                        1313       namesleft #1 >
                        1314       { ", " * t * }
                        1315       {
                        1316         numnames #2 >
                        1317         curlanguage "english" =
                        1318         and
                        1319         { ", " * }
                        1320         'skip$
                        1321         if$
                        1322         t "others" =
                        1323         %t "~others" =
                        1324         %or
                        1325         %{ " et~al." * }
                        1326         { " " bbl.etal * * }
                        1327         %{ " and " * t * }
                        1328         { " " bbl.and " " * * * t * }
                        1329         if$
                        1330       }
                        1331       if$
                        1332     }
                        1333     't
                        1334     if$
                        1335     nameptr #1 + 'nameptr :=
                        1336     namesleft #1 - 'namesleft :=
                        1337   }
                        1338   while$
                        1339 }
1340
1341 FUNCTION {author.full}
1342 { author empty$

```

```

1343     { "" }
1344     { author format.full.names }
1345     if$
1346 }
1347
1348 FUNCTION {editor.full}
1349 { editor empty$
1350     { "" }
1351     { editor format.full.names }
1352     if$
1353 }
1354
1355 FUNCTION {author.editor.full}
1356 { author empty$
1357     { editor empty$
1358         { "" }
1359         { editor format.full.names }
1360         if$
1361     }
1362     { author format.full.names }
1363     if$
1364 }
1365
1366 FUNCTION {make.full.names}
1367 { type$ "book" =
1368     type$ "inbook" =
1369     or
1370     'author.editor.full
1371     { type$ "proceedings" =
1372         'editor.full
1373         'author.full
1374         if$
1375     }
1376     if$
1377 }
1378
1379 % =====
1380 FUNCTION {output.bibitem}
1381 { newline$
1382     "\bibitem[" write$
1383     label write$
1384     ")" make.full.names duplicate$ short.list =
1385         { pop$ }
1386         { * }
1387         if$
1388     "]" * write$
1389     cite$ write$
1390     "]" write$
1391 %% language empty$
1392 %%     { "english" 'curlanguage := }

```

```

1393 %% {language 'curlanguage := }
1394 %% if$
1395 langid empty$
1396 { language empty$
1397 { "english" 'curlanguage := }
1398 { language 'curlanguage := }
1399 if$
1400 }
1401 { langid 'curlanguage := }
1402 if$
1403 "\selectlanguageifdefined" curlanguage bracyfy * write$
1404 newline$
1405 ""
1406 before.all 'output.state :=
1407 }
1408 % =====
1409 %FUNCTION {output.bibitem}
1410 %{ newline$
1411 % "\bibitem" write$
1412 %% author.key.label
1413 %% year parenthesify *
1414 %% "; lbl:" label * *
1415 %% "; mfn:" make.full.names * *
1416 % label
1417 % make.full.names *
1418 % bracketise write$
1419 % cite$ bracyfy write$
1420 % newline$
1421 % language empty$
1422 % { "english" 'curlanguage := }
1423 % {language 'curlanguage := }
1424 % if$
1425 % "\selectlanguageifdefined" curlanguage bracyfy * write$
1426 % newline$
1427 % ""
1428 % before.all 'output.state :=
1429 %}
1430 % =====
1431 </natbib>
1432

```

n.dashify

```

1433 FUNCTION {n.dashify}
1434 { 't :=
1435 ""
1436 { t empty$ not }
1437 { t #1 #1 substring$ "-" =
1438 { t #1 #2 substring$ "--" = not
1439 { "--" *
1440 t #2 global.max$ substring$ 't :=
1441 }

```

```

1442         { { t #1 #1 substring$ "-" = }
1443         { "-" *
1444           t #2 global.max$ substring$ 't :=
1445         }
1446         while$
1447       }
1448     if$
1449   }
1450   { t #1 #1 substring$ *
1451     t #2 global.max$ substring$ 't :=
1452   }
1453   if$
1454 }
1455 while$
1456 }
1457

```

word.in

```

1458 FUNCTION {word.in}
1459 { bbl.iin
1460   " " * }
1461

```

format.btitle

```

1462 FUNCTION {format.btitle}
1463 { title
1464 }
1465

```

.or.space.connect

```

1466 FUNCTION {tie.or.space.connect}
1467 { duplicate$ text.length$ #3 <
1468   { "~" }
1469   { " " }
1470   if$
1471   swap$ * *
1472 }
1473

```

tie.connect    Declare function to insert unbreakable space before last word in stack.

```

1474 FUNCTION {tie.connect}
1475 {"~"
1476   swap$ * *
1477 }
1478
1479

```

either.or.chec

```

1480 FUNCTION {either.or.check}
1481 { empty$
1482   'pop$
1483   { "can't use both " swap$ * " fields in " * cite$ * warning$ }

```

```

1484 if$
1485 }
1486

```

format.bvolume

```

1487 FUNCTION {format.bvolume}
1488 { volume empty$
1489   { "" }
1490   { bbl.vvol volume tie.connect
1491     series empty$
1492     'skip$
1493     { bbl.of space.word * series emphasize * }
1494     if$
1495     "volume and number" number either.or.check
1496   }
1497   if$
1498 }
1499

```

mat.number.series

```

1500 FUNCTION {format.number.series}
1501 { volume empty$
1502   { number empty$
1503     { series field.or.null }
1504     { series empty$
1505       { "there's a number but no series in " cite$ * warning$
1506         bbl.nnr }
1507       {
1508         %new.dblslash
1509         new.sentence
1510         series
1511         bbl.nr
1512         tie.or.space.connect}
1513       if$
1514       number tie.or.space.connect
1515     }
1516     if$
1517   }
1518   { "" }
1519   if$
1520 }
1521

```

is.num

```

1522 FUNCTION {is.num}
1523 { chr.to.int$
1524   duplicate$ "0" chr.to.int$ < not
1525   swap$ "9" chr.to.int$ > not and
1526 }
1527

```

extract.num



```

1528 FUNCTION {extract.num}
1529 { duplicate$ 't :=
1530   "" 's :=
1531   { t empty$ not }
1532   { t #1 #1 substring$
1533     t #2 global.max$ substring$ 't :=
1534     duplicate$ is.num
1535     { s swap$ * 's := }
1536     { pop$ "" 't := }
1537     if$
1538   }
1539   while$
1540   s empty$
1541   'skip$
1542   { pop$ s }
1543   if$
1544 }
1545
1546 <*debug>

```

eng.ord

```

1547 FUNCTION {eng.ord}
1548 { duplicate$ "1" swap$ *
1549   #-2 #1 substring$ "1" =
1550     { bbl.th * }
1551     { duplicate$ #-1 #1 substring$
1552       duplicate$ "1" =
1553         { pop$ bbl.st * }
1554         { duplicate$ "2" =
1555           { pop$ bbl.nd * }
1556           { "3" =
1557             { bbl.rd * }
1558             { bbl.th * }
1559             if$
1560           }
1561           if$
1562         }
1563         if$
1564       }
1565     if$
1566 }
1567 </debug>
1568

```

convert.edition

```

1569 FUNCTION {convert.edition}
1570 { edition
1571 % edition extract.num "1" change.case$ 's :=
1572 % s "first" = s "1" = or
1573 %   { bbl.first 't := }
1574 %   { s "second" = s "2" = or

```

```

1575 %      { bbl.second 't := }
1576 %      { s "third" = s "3" = or
1577 %          { bbl.third 't := }
1578 %          { s "fourth" = s "4" = or
1579 %              { bbl.fourth 't := }
1580 %              { s "fifth" = s "5" = or
1581 %                  { bbl.fifth 't := }
1582 %                  { s #1 #1 substring$ is.num
1583 %                      { s eng.ord 't := }
1584 %                      { edition 't := }
1585 %                  if$
1586 %              }
1587 %          if$
1588 %      }
1589 %      if$
1590 %  }
1591 %      if$
1592 %  }
1593 %      if$
1594 %  }
1595 %  if$
1596 %  t
1597 }
1598

```

format.edition

```

1599 FUNCTION {format.edition}
1600 { edition empty$
1601     { "" }
1602     { output.state mid.sentence =
1603         { convert.edition "l" change.case$ " " * bbl.edition * }
1604         { convert.edition "t" change.case$ " " * bbl.edition * }
1605         { convert.edition " " * bbl.edition * }
1606         { convert.edition " " * bbl.edition * }
1607     if$
1608 }
1609 if$
1610 }
1611
1612 INTEGERS { multiresult }
1613

```

multi.page.check

```

1614 FUNCTION {multi.page.check}
1615 { 't :=
1616   #0 'multiresult :=
1617   { multiresult not
1618     t empty$ not
1619     and
1620   }
1621   { t #1 #1 substring$

```

```

1622     duplicate$ "-" =
1623     swap$ duplicate$ "," =
1624     swap$ "+" =
1625     or or
1626     { #1 'multiresult := }
1627     { t #2 global.max$ substring$ 't := }
1628     if$
1629     }
1630 while$
1631 multiresult
1632 }
1633

```

format.pages

```

1634 %%FUNCTION {format.pages}
1635 %%{ pages empty$
1636 %%  { "" }
1637 %%  { pages multi.page.check
1638 %%    { bbl.ppages pages n.dashify tie.connect }
1639 %%    { bbl.ppage pages tie.connect }
1640 %%  if$
1641 %%  }
1642 %%  if$
1643 %%}
1644 FUNCTION {format.pages}
1645 { eid empty$
1646   {
1647     pages empty$
1648     { "" }
1649     { pages multi.page.check
1650       { bbl.ppages pages n.dashify tie.connect }
1651       { bbl.ppage pages tie.connect }
1652     if$
1653     }
1654     if$
1655   }
1656   { eid multi.page.check
1657     { bbl.ppages eid n.dashify tie.connect }
1658     { bbl.ppage eid tie.connect }
1659   if$
1660   }
1661   if$
1662 }
1663

```

format.pages.page

```

1664 %%FUNCTION {format.pages.page}
1665 %%{ pages empty$
1666 %%  { pagetotal empty$
1667 %%  { "" }
1668 %%  { pagetotal bbl.pages tie.connect }

```

```

1669 %%      if$}
1670 %%      { format.pages}
1671 %%      if$
1672 %%}
1673 FUNCTION {format.pages.page}
1674 { eid empty$
1675   { pages empty$
1676     { pagetotal empty$
1677       { "" }
1678       { pagetotal bbl.pages tie.connect }
1679       if$
1680     }
1681     { format.pages}
1682     if$
1683   }
1684   { format.pages }
1685   if$
1686 }
1687

```

mat.vol.num.pages

```

1688 FUNCTION {format.vol.num.pages}
1689 { volume field.or.null
1690   number empty$
1691   'skip$
1692   {
1693     ", no." number tie.or.space.connect *
1694     volume empty$
1695     { "there's a number but no volume in " cite$ * warning$ }
1696     'skip$
1697     if$
1698   }
1699   if$
1700   pages empty$
1701   'skip$
1702   { duplicate$ empty$
1703     { pop$ format.pages }
1704     { ": " * pages n.dashify * }
1705     if$
1706   }
1707   if$
1708 }
1709

```

format.volume

```

1710 FUNCTION {format.volume}
1711 { volume empty$
1712   { "" }
1713   { bbl.vvol volume tie.or.space.connect }
1714   if$
1715 }

```

```

1716
format.number
1717 FUNCTION {format.number}
1718 { number empty$
1719   { "" }
1720   { bbl.nr number tie.or.space.connect }
1721   if$
1722 }
1723
1724 <*debug>

mat.chapter.pages
1725 FUNCTION {format.chapter.pages}
1726 { chapter empty$
1727   'format.pages
1728   { type empty$
1729     { bbl.chapter }
1730     { type "1" change.case$ }
1731     if$
1732     chapter tie.or.space.connect
1733     pages empty$
1734     'skip$
1735     { ", " * format.pages * }
1736     if$
1737   }
1738   if$
1739 }
1740 </debug>
1741

empty.misc.check
1742 FUNCTION {empty.misc.check}
1743 { author empty$ title empty$ howpublished empty$
1744   month empty$ year empty$ note empty$
1745   and and and and and
1746   key empty$ not and
1747   { "all relevant fields are empty in " cite$ * warning$ }
1748   'skip$
1749   if$
1750 }
1751

ormat.thesis.type
1752 FUNCTION {format.thesis.type}
1753 { type empty$
1754   'skip$
1755   { pop$
1756     <!utf8>      type "t" change.case$
1757     <utf8>      type
1758   }
1759   if$

```

```
1760 }
1761
```

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

```
1762 %FUNCTION {format.techrep.type.number}
1763 %{ type empty$
1764 %   { bbl.techreport }
1765 %   'type
1766 %   if$
1767 %   number empty$
1768 %<utf8> { "t" change.case$ }
1769 %<utf8> { "" }
1770 %   { number tie.or.space.connect }
1771 %   if$
1772 %}
1773
1774 FUNCTION {format.techreport.type}
1775 { type empty$
1776   { bbl.techreport }
1777   'type
1778   if$
1779 }
1780
```

**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**.

```
1781 <!!long>
1782 FUNCTION {author.before}
1783 {
1784   author empty$
1785   'skip$
1786   {author num.names$ #4 <
1787     {format.authors output
1788       new.sentence}
1789     'skip$
1790     if$}
1791   if$
1792 }
1793 </!!long>
```

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

```
1794 <!!long>
1795 FUNCTION {author.before}
```

```

1796 {
1797   author empty$
1798   'skip$
1799   { format.authors output
1800     new.sentence
1801   }
1802   if$
1803 }
1804 </long>
1805

```

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

```

1806 %%<!*!long>
1807 %%FUNCTION {bookauthor.before}
1808 %%{
1809 %%  bookauthor empty$
1810 %%    'skip$
1811 %%    {bookauthor num.names$ #4 <
1812 %%      {format.bookauthors output
1813 %%        new.sentence}
1814 %%    'skip$
1815 %%    if$}
1816 %%  if$
1817 %%}
1818 %%</!*!long>
1819 %%<!*long>
1820 %%FUNCTION {bookauthor.before}
1821 %%{
1822 %%  bookauthor empty$
1823 %%    'skip$
1824 %%    { format.bookauthors output
1825 %%      new.sentence
1826 %%    }
1827 %%  if$
1828 %%}
1829 %%</long>
1830

```

`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).

```

1831 <!*!modern>
1832 FUNCTION {author.after}
1833 {
1834   author empty$
1835   'skip$
1836   {format.authors.after output
1837     new.semicolon }

```

```

1838 if$
1839 }
1840 </!modern>
1841 <*modern>
1842 <*!long>
1843 FUNCTION {author.after}
1844 {
1845   author empty$
1846   'skip$
1847   {author num.names$ #3 >
1848     {format.authors.after output
1849       new.semicolon }
1850   'skip$
1851   if$}
1852 if$
1853 }
1854 </!long>
1855 <*long>
1856 FUNCTION {author.after} { }
1857 </long>
1858 </modern>
1859

```

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

```

1860 FUNCTION {bookauthor.after}
1861 {
1862   bookauthor empty$
1863   'skip$
1864   {format.bookauthors.after output
1865     new.semicolon }
1866 if$
1867 }
1868

```

organization.after

```

1869 FUNCTION {editor.organization.after}
1870 {
1871   compiler empty$
1872   {}
1873   { format.compiler.after output
1874     new.semicolon
1875   }
1876 if$
1877 editor empty$
1878   {}
1879   { format.editors.after output
1880     new.semicolon
1881   }
1882 if$
1883 organization empty$

```



```

1884     {}
1885     {organization output
1886     new.semicolon
1887     }
1888     if$
1889 }
1890
format.url
1891 FUNCTION {format.url}
1892 { url empty$
1893   { "" }
1894   {
1895     !(modern|strict)      "\BibUrl{ " url * "}" *
1896     (modern|strict)      bbl.url ": \BibUrl{" * url * "}" *
1897     urldate empty$
1898     { "" }
1899     { " (" bbl.urldate * ": " * urldate * ")" * }
1900     if$ *
1901   }
1902   if$
1903 }
1904
output.url
1905 FUNCTION {output.url}
1906 {
1907   url empty$
1908   'skip$
1909   { format.url output }
1910   if$
1911 }
1912
format.annotate
1913 FUNCTION {format.annotate}
1914 { annotate empty$
1915   { "" }
1916   { after.sentence 'output.state :=
1917     "\BibAnnote{" annotate add.period$ * "}" *
1918   }
1919   if$
1920 }
1921
format.isbn    Do we really need to provide electronic search for ISBN?
1922 FUNCTION {format.isbn}
1923 {
1924   isbn empty$
1925   { "" }
1926   { "ISBN:~\href{http://isbndb.com/search-all.html?kw=" isbn *
1927     "}{ " * isbn * "}" *

```

```

1928     }
1929   if$
1930 }
1931

```

**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 identifier at <http://dx.doi.org/> web-site.

```

1932 < *eprint >
1933 FUNCTION {add.doi}
1934 { duplicate$ empty$
1935   'skip$
1936   { doi empty$
1937     'skip$
1938     { "\href{http://dx.doi.org/" doi * "}{" * swap$ * "}" * }
1939     if$
1940   }
1941   if$
1942 }
1943 < /eprint >

```

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

```

1944 < *!eprint >
1945 FUNCTION {add.doi} { }
1946 < /!eprint >
1947

```

**add.media** New in version 2. Adds media field if **strict** options is in effect.

```

1948 < *!strict >
1949 FUNCTION {add.media} { }
1950 < /!strict >
1951 < *strict >
1952 FUNCTION {add.media}
1953 { duplicate$ empty$
1954   'skip$
1955   { media empty$
1956     { " " * bbl.media enclose.square.brackets * }
1957     { " " * media enclose.square.brackets * }
1958 %%     { bbl.media enclose.square.brackets * }
1959 %%     { media enclose.square.brackets * }
1960     if$
1961   }
1962   if$
1963 }
1964 < /strict >
1965

```

## 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. 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.

```
1966 <*eprint>
1967 %FUNCTION {format.eprint}
1968 %{ eprint empty$
1969 %   { "" }
1970 %   { eprintclass empty$
1971 %     { " \href{http://arxiv.org/abs/" eprint * "}" * "{" * eprint * "}" * }
1972 %     { eprinttype empty$
1973 %       { " \href{http://arxiv.org/abs/" eprint * "}" *
1974 %         {" * eprintclass * "/" * eprint * "}" *
1975 %       }
1976 %       { " \href{http://arxiv.org/abs/" eprint * "}" *
1977 %         {" * eprinttype * ":" * eprintclass * "/" * eprint * "}" *
1978 %       }
1979 %     if$}
1980 %   if$}
1981 %if$}
1982
1983 %FUNCTION {format.eprint}
1984 %{ eprint empty$
1985 %   { "" }
1986 %   { eprinttype empty$
1987 %     { "" }
1988 %     { eprinttype "~: " *}
1989 %   if$
1990 %     eprintclass empty$
1991 %     { }
1992 %     { eprintclass * "/" *}
1993 %   if$
1994 %     eprint *
1995 %   }
1996 % if$
1997 % url empty$
1998 %   { }
```

```

1999 % { "\href{" url * "}{ " * swap$ * "}" *}
2000 % if$
2001 %}
2002
2003 FUNCTION {format.eprint}
2004 { eprint empty$
2005   { "" }
2006   { eprinttype empty$
2007     { "" }
2008     { eprinttype "~: " *}
2009     if$
2010     eprintclass empty$
2011     { }
2012     { eprintclass * "/" *}
2013     if$
2014     url empty$
2015     { eprint * }
2016     { "\href{" * url * "}{ " * eprint * "}" *}
2017     if$
2018   }
2019   if$
2020 }
2021
2022 FUNCTION {output.eprint.url}
2023 {
2024   eprint empty$
2025   { format.url output }
2026   { format.eprint output }
2027   if$
2028 }
2029
2030 </eprint>
2031
2032 <!*eprint>
2033 FUNCTION {output.eprint.url}
2034 {
2035   format.url output
2036 }
2037 </!*eprint>
2038

```

Functions added in v1.2f to format patent entry (thanks to Stanislav Kruchinin).

add.number

```

2039 FUNCTION {add.number}
2040 { duplicate$ empty$
2041   { "" }
2042   { bbl.nr swap$ tie.or.space.connect }
2043   if$
2044 }
2045

```

ormat.type.number

```
2046 FUNCTION {format.type.number}
2047 {
2048   type empty$
2049   { "" }
2050   {
2051     number empty$
2052     { "" }
2053     { type number tie.or.space.connect }
2054     if$
2055   }
2056   if$
2057 }
2058
```

ormat.requestdate

```
2059 FUNCTION {format.requestdate}
2060 { requestdate empty$
2061   { "" }
2062   { bbl.req requestdate tie.or.space.connect }
2063   if$
2064 }
2065
```

t.publicationdate

```
2066 FUNCTION {format.publicationdate}
2067 { publicationdate empty$
2068   { "" }
2069   { bbl.publ publicationdate tie.or.space.connect }
2070   if$
2071 }
2072
```

ormat.prioritydate

```
2073 FUNCTION {format.prioritydate}
2074 { prioritydate empty$
2075   { "" }
2076   { bbl.priority prioritydate tie.or.space.connect }
2077   if$
2078 }
2079
```

## 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`.

```
2080 FUNCTION {article}
2081 {
2082   output.bibitem
2083   author.before
2084   (natbib) author format.key output
2085   format.title add.media "title" output.check
2086   new.slash
2087   author.after
2088   new.dblslash
2089   journal emphasize add.doi "journal" output.check % new in v.2
2090   new.block
2091   format.date "year" output.check
2092   new.block
2093   format.volume output
2094   format.number output
2095   new.block
2096   format.pages.page output
2097   new.block
2098   note output
2099   new.sentence
2100   % format.url output
2101   output.eprint.url
2102   format.annotate output
2103   fin.entry
2104 }
2105
```

**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.

```
2106 FUNCTION {book}
2107 {
2108   output.bibitem
2109   author.before
2110   (natbib) author format.key output
2111   format.btitle add.doi add.media "title" output.check
2112   new.slash
2113   author.after
2114   editor.organization.after
2115   new.sentence
2116   format.number.series output
2117   new.block
2118   format.edition output
2119   new.block
2120   output.address.publisher
2121   format.date "year" output.check
2122   new.block
2123   format.bvolume output
2124   new.block
2125   format.pages.page output
2126   new.block
2127   (eprint) format.isbn output
2128   (eprint) new.block
2129   note output
2130   new.sentence
2131   % format.url output
2132   output.eprint.url
2133   format.annotate output
2134   fin.entry
2135 }
2136
```

**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.

```
2137 FUNCTION {booklet}
2138 {
2139   output.bibitem
2140   author.before
2141   (natbib) author format.key output
2142   format.title add.doi add.media "title" output.check
2143   new.slash
2144   author.after
2145   editor.organization.after
2146   new.block
2147   howpublished output

```

```

2148 address output
2149 format.date "year" output.check
2150 new.block
2151 note output
2152 new.sentence
2153 % format.url output
2154 output.eprint.url
2155 format.annotate output
2156 fin.entry
2157 }
2158

```

**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.

```

2159 FUNCTION {inbook}
2160 {
2161   output.bibitem
2162   author.before
2163   (natbib) author format.key output
2164   format.btitle add.doi add.media "title" output.check
2165   new.slash
2166   author.after
2167   new.dblslash
2168   % bookauthor.before
2169   booktitle "booktitle" output.check
2170   new.slash
2171   bookauthor.after
2172   editor.organization.after
2173   new.block
2174   format.edition output
2175   new.block
2176   format.number.series output
2177   new.sentence
2178   output.address.publisher
2179   format.date "year" output.check
2180   new.block
2181   format.bvolume output
2182   new.block
2183   format.pages.page output
2184   new.block
2185   (eprint) format.isbn output
2186   (eprint) new.block
2187   note output
2188   new.sentence

```



```

2189 % format.url output
2190 output.eprint.url
2191 format.annotate output
2192 fin.entry
2193 }
2194

```

**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.

```

2195 FUNCTION {incollection}
2196 {
2197   output.bibitem
2198   author.before
2199   (natbib) author format.key output
2200   new.sentence
2201   format.title add.doi add.media "title" output.check
2202   new.slash
2203   author.after
2204   new.dblslash
2205   booktitle "booktitle" output.check
2206   new.slash
2207   editor.organization.after
2208   new.block
2209   output.address.publisher
2210   format.date "year" output.check
2211   new.block
2212   format.bvolume output
2213   format.number.series output
2214   new.block
2215   format.pages.page output
2216   new.block
2217   note output
2218   new.sentence
2219 % format.url output
2220 output.eprint.url
2221 format.annotate output
2222 fin.entry
2223 }
2224

```

**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.

```

2225 FUNCTION {proceedings}
2226 {
2227   output.bibitem
2228   (natbib) editor format.key output
2229   format.btitle add.doi add.media "title" output.check
2230   new.slash
2231   editor.organization.after
2232   new.block
2233   format.bvolume output
2234   format.number.series output
2235   % address empty$
2236   %   { publisher output
2237   %     format.date "year" output.check
2238   %   }
2239   %   { address output.nonnull
2240   %     format.date "year" output.check
2241   %     new.sentence
2242   %     publisher output
2243   %   }
2244   % if$
2245   output.address.publisher
2246   format.date "year" output.check
2247   new.block
2248   note output
2249   new.sentence
2250   % format.url output
2251   output.eprint.url
2252   format.annotate output
2253   fin.entry
2254 }
2255

```

**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.

```

2256 FUNCTION {inproceedings}
2257 { output.bibitem
2258   author.before
2259   (natbib) author format.key output
2260   new.sentence
2261   format.title add.doi add.media "title" output.check
2262   new.slash
2263   author.after

```

```

2264 new.dblslash
2265 booktitle "booktitle" output.check
2266 new.slash
2267 editor.organization.after
2268 new.block
2269 format.bvolume output
2270 format.number.series output
2271 new.block
2272 % address empty$
2273 % { publisher output
2274 %   format.date "year" output.check
2275 % }
2276 % { address output.nonnull
2277 %   new.colon
2278 %   publisher output
2279 %   format.date "year" output.check
2280 % }
2281 % if$
2282 output.address.publisher
2283 format.date "year" output.check
2284 new.block
2285 format.pages.page output
2286 new.block
2287 note output
2288 new.sentence
2289 % format.url output
2290 output.eprint.url
2291 format.annotate output
2292 fin.entry
2293 }
2294

```

**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.

```

2295 FUNCTION {manual}
2296 { output.bibitem
2297   author empty$
2298     { organization empty$
2299       'skip$
2300       { organization output.nonnull
2301         address output
2302       }
2303     if$
2304   }
2305   { format.authors output.nonnull }
2306   if$
2307   (natbib) author format.key output

```

```

2308 new.block
2309 format.btitle add.doi add.media "title" output.check
2310 author empty$
2311 { organization empty$
2312 {
2313     address new.block.checka
2314     address output
2315 }
2316     'skip$
2317 if$
2318 }
2319 {
2320     organization address new.block.checkb
2321     organization output
2322     address output
2323 }
2324 if$
2325 format.edition output
2326 format.date "year" output.check
2327 new.block
2328 note output
2329 new.sentence
2330 % format.url output
2331 output.eprint.url
2332 format.annotate output
2333 fin.entry
2334 }
2335

```

**patent** A patent or patent request. The number or record token is given in the number field. Use the **type** field to specify the type and the **location** field to indicate the scope of the patent, if different from the scope implied by the type. Note that the location field is treated as a key list with this entry type.

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

Optional fields: holder, subtitle, titleaddon, type, version, location, note, date, month, year, addendum, pubstate, doi, eprint, eprint class, eprint type, url, urldate.

```

2336 FUNCTION {patent}
2337 {
2338     output.bibitem
2339 % title output.nonnull
2340 title add.media output.nonnull
2341 % add.blank
2342 % media enclose.square.brackets output % S.Kruchinin's version
2343 new.colon
2344 format.type.number output
2345 add.blank
2346 location output
2347 new.colon
2348 ipc output
2349 new.slash

```

```

2350 format.authors.after "author" output.check
2351 add.blank
2352 authorcountry enclose.round.brackets output.nonnull
2353 (natbib) author format.key output
2354 new.semicolon
2355 holder output.nonnull
2356 new.semicolon
2357 credits output.nonnull
2358 new.block
2359 requestnumber add.number output
2360 new.semicolon
2361 format.requestdate output
2362 new.semicolon
2363 format.publicationdate output
2364 publication output
2365 new.semicolon
2366 format.prioritydate output
2367 prioritynumber output
2368 prioritycountry enclose.round.brackets output
2369 new.block
2370 note output
2371 new.sentence
2372 output.url
2373 format.annotate output
2374 new.block
2375 pagetotal output
2376 fin.entry
2377 }
2378

```

**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`.

```

2379 FUNCTION {misc}
2380 { output.bibitem
2381   format.authors output
2382   (natbib) author format.key output
2383   title howpublished new.sentence.checkb
2384   format.title add.media output
2385   howpublished new.block.checka
2386   howpublished output
2387   new.block
2388   format.date "year" output.check
2389   new.block
2390   note output
2391   new.sentence
2392 %   format.url output
2393   output.eprint.url
2394   format.annotate output

```

```

2395  fin.entry
2396 }
2397

```

**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

```

2398 FUNCTION {unpublished}
2399 { output.bibitem
2400   author.before
2401   (natbib)  author format.key output
2402   format.btitle "title" output.check
2403   new.slash
2404   author.after
2405   editor.organization.after
2406   new.block
2407   format.date "year" output.check
2408   new.block
2409   note "note" output.check
2410   new.sentence
2411   % format.url output
2412   output.eprint.url
2413   format.annote output
2414   fin.entry
2415 }
2416

```

**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.

```

2417 FUNCTION {online}
2418 { output.bibitem
2419   format.authors output
2420   (natbib)  author format.key output
2421   title howpublished new.sentence.checkb
2422   format.title add.doi add.media "title" output.check
2423   % howpublished new.block.checka
2424   howpublished new.dblslash.checka
2425   (!strict) howpublished output
2426   (strict)  howpublished enclose.square.brackets output
2427   editor.organization.after
2428   new.sentence
2429   new.block

```

```

2430 output.address.publisher
2431 format.date output
2432 new.block
2433 % format.url output
2434 output.eprint.url
2435 new.sentence
2436 note output
2437 format.annotate output
2438 fin.entry
2439 }
2440
internet      New in version 2012.02.15.
  www 2441 FUNCTION {internet}    {online}
  webpage 2442 FUNCTION {www}      {online}
  ielectronic 2443 FUNCTION {webpage} {online}
            2444 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

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

```

```

2472 new.sentence
2473 format.btitle add.doi add.media "title" output.check
2474 new.colon
2475 % bbl.phdthesis format.thesis.type output.nonnull
2476 type "type" output.check
2477 new.colon
2478 number output
2479 new.slash
2480 format.authors.after output
2481 new.semicolon
2482 school "school" output.check
2483 new.block
2484 output.address.publisher
2485 format.date "year" output.check
2486 new.block
2487 format.pages.page output
2488 new.block
2489 note output
2490 new.sentence
2491 % format.url output
2492 output.eprint.url
2493 format.annotate output
2494 fin.entry
2495 }
2496 </modern | strict>
2497

```

**report**      New in version 2012.02.02.

A technical report, research report, or white paper published by a university or some other institution. Use the type field to specify the type of report. The sponsoring institution goes in the institution field.

Required fields: author, title, type, institution, year/date.

Optional fields: subtitle, titleaddon, language, number, version, note, location, month, isrn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```

2498 %FUNCTION {report}
2499 %{
2500 %   output.bibitem
2501 %   author.before
2502 %   new.sentence
2503 %   format.title add.doi add.media "title" output.check
2504 %   new.colon
2505 %%   format.techrep.type.number output.nonnull
2506 %   type "type" output.check
2507 %   new.slash
2508 %   author.after
2509 %   editor.organization.after
2510 %   new.block
2511 %   address output
2512 %   new.colon
2513 %   institution "institution" output.check

```



```

2514 % format.date "year" output.check
2515 % new.block
2516 % note output
2517 % new.block % v.2
2518 % format.pages.page output % v.2
2519 % new.sentence
2520 %% format.url output
2521 % output.eprint.url
2522 % format.annotate output
2523 % fin.entry
2524 %}
2525 FUNCTION {report}
2526 {
2527   output.bibitem
2528 %   author.before
2529 %   new.sentence
2530   format.title add.doi add.media "title" output.check
2531   (natbib) title format.key output
2532   new.colon
2533 %   format.techrep.type.number output.nonnull
2534 %   type "type" output.check
2535 %   format.report.type.number "type" output.check
2536   type "type" output.check
2537   new.colon
2538   number output
2539   new.slash
2540   institution "institution" output.check
2541   new.semicolon
2542   format.chief.after output % from editor field
2543   new.semicolon
2544   format.executor.after output % from author field
2545   new.block
2546   address output
2547   new.colon
2548   organization output
2549   format.date "year" output.check
2550   new.block % v.2
2551   format.pages.page output % v.2
2552   new.sentence % или new.block ?
2553   output.eprint.url
2554   new.block
2555   note output
2556   format.annotate output
2557   fin.entry
2558 }
2559

```

### 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.

```
2560 <*(modern|strict)>
2561 FUNCTION {phdthesis}
2562 { output.bibitem
2563   format.authors "author" output.check
2564   new.sentence
2565   format.btitle "title" output.check
2566   new.colon
2567   bbl.phdthesis format.thesis.type output.nonnull
2568   new.slash
2569   school "school" output.check
2570   new.block
2571   output.address.publisher.date
2572   new.block
2573   format.pages.page output
2574   note output
2575   new.sentence
2576   format.url output
2577   format.annotate output
2578   fin.entry
2579 }
2580 </!(modern|strict)>
2581 <*(modern|strict)>
2582 FUNCTION {phdthesis}
2583 { output.bibitem
2584   format.authors "author" output.check
2585   natbib author format.key output
2586   new.sentence
2587   format.btitle add.doi add.media "title" output.check
2588   new.colon
2589   bbl.phdthesis format.thesis.type output.nonnull
2590   new.colon
2591   number output
2592   new.slash
2593   format.authors.after output
2594   new.semicolon
2595   school "school" output.check
2596   new.block
2597   output.address.publisher
2598   format.date "year" output.check
2599   new.block
2600   format.pages.page output
2601   new.block
```

```

2602  note output
2603  new.sentence
2604 %  format.url output
2605  output.eprint.url
2606  format.annotate output
2607  fin.entry
2608 }
2609 </modern | strict>
2610

```

**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.

```

2611 <*(modern | strict)>
2612 FUNCTION {mastersthesis}
2613 { output.bibitem
2614   format.authors "author" output.check
2615 <natbib> author format.key output
2616   new.sentence
2617   format.btitle "title" output.check
2618   new.colon
2619   bbl.mthesis format.thesis.type output.nonnull
2620   new.slash
2621   school "school" output.check
2622   new.block
2623   output.address.publisher.date
2624   new.block
2625   format.pages.page output
2626   note output
2627   new.sentence
2628   format.url output
2629   format.annotate output
2630   fin.entry
2631 }
2632 </!(modern | strict)>
2633 <*(modern | strict)>
2634 FUNCTION {mastersthesis}
2635 { output.bibitem
2636   format.authors "author" output.check
2637 <natbib> author format.key output
2638   new.sentence
2639   format.btitle add.doi add.media "title" output.check
2640   new.colon
2641   bbl.mthesis format.thesis.type output.nonnull
2642   new.colon
2643   number output
2644   new.slash
2645   format.authors.after output
2646   new.semicolon
2647   school "school" output.check
2648   new.block
2649   output.address.publisher

```

```

2650  format.date "year" output.check
2651  new.block
2652  format.pages.page output
2653  new.block
2654  note output
2655  new.sentence
2656 %  format.url output
2657  output.eprint.url
2658  format.annotate output
2659  fin.entry
2660 }
2661 </modern | strict>
2662

```

**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.

```

2663 <*(modern | strict)>
2664 FUNCTION {dscithesis}
2665 { output.bibitem
2666   format.authors "author" output.check
2667 <natbib> author format.key output
2668   new.sentence
2669   format.btitle "title" output.check
2670   new.colon
2671   bbl.dscithesis format.thesis.type output.nonnull
2672   new.slash
2673   school "school" output.check
2674   new.block
2675   output.address.publisher.date
2676   new.block
2677   format.pages.page output
2678   note output
2679   new.sentence
2680   format.url output
2681   format.annotate output
2682   fin.entry
2683 }
2684 </!(modern | strict)>
2685 <*modern | strict>
2686 FUNCTION {dscithesis}
2687 { output.bibitem
2688   format.authors "author" output.check
2689 <natbib> author format.key output
2690   new.sentence
2691   format.btitle add.doi add.media "title" output.check
2692   new.colon
2693   bbl.dscithesis format.thesis.type output.nonnull
2694   new.colon
2695   number output
2696   new.slash
2697   format.authors.after output

```

```

2698 new.semicolon
2699 school "school" output.check
2700 new.block
2701 output.address.publisher
2702 format.date "year" output.check
2703 new.block
2704 format.pages.page output
2705 new.block
2706 note output
2707 new.sentence
2708 % format.url output
2709 output.eprint.url
2710 format.annotate output
2711 fin.entry
2712 }
2713 </modern | strict>
2714

```

conference

```

2715 FUNCTION {conference} { inproceedings }
2716

```

**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.

```

2717 %FUNCTION {techreport}
2718 %{
2719 % output.bibitem
2720 % author.before
2721 % new.sentence
2722 % format.title add.doi add.media "title" output.check
2723 % new.colon
2724 % format.techrep.type.number output.nonnull
2725 % new.slash
2726 % author.after
2727 % editor.organization.after
2728 % new.block
2729 % address output
2730 % new.colon
2731 % institution "institution" output.check
2732 % format.date "year" output.check
2733 % new.block
2734 % note output
2735 % new.block % v.2
2736 % format.pages.page output % v.2
2737 % new.sentence
2738 %% format.url output
2739 % output.eprint.url
2740 % format.annotate output
2741 % fin.entry
2742 %}
2743

```

```

2744 FUNCTION {techreport}
2745 {
2746   output.bibitem
2747   %   author.before
2748   %   new.sentence
2749   format.title add.doi add.media "title" output.check
2750   (natbib)   title format.key output
2751   new.colon
2752   %   format.techrep.type.number output.nonnull
2753   %   type "type" output.check
2754   %   format.report.type.number "type" output.check
2755   %   type output
2756   format.techreport.type output
2757   new.colon
2758   number output
2759   new.slash
2760   institution "institution" output.check
2761   new.semicolon
2762   format.chief.after output % from editor field
2763   new.semicolon
2764   format.executor.after output % from author field
2765   new.block
2766   address output
2767   new.colon
2768   organization output
2769   format.date "year" output.check
2770   new.block           % v.2
2771   format.pages.page output % v.2
2772   new.sentence % или new.block ?
2773   output.eprint.url
2774   new.block
2775   note output
2776   format.annotate output
2777   fin.entry
2778 }
2779
2780

```

default.type

```

2781 FUNCTION {default.type} { misc }
2782

```

## 2.7 Month Abbreviations

Borrowed from `merlin.mbs` of package `custom-bib`. This is done for backward compatibility with standard `.bst` styles which are designed for English. The string in the definition of any month macro must coincide with that used in `format.month` function in the above.

```

2783 MACRO {jan} {"Jan."}
2784 MACRO {feb} {"Feb."}
2785 MACRO {mar} {"Mar."}

```

```

2786 MACRO {apr} {"Apr."}
2787 MACRO {may} {"May"}
2788 MACRO {jun} {"Jun."}
2789 MACRO {jul} {"Jul."}
2790 MACRO {aug} {"Aug."}
2791 MACRO {sep} {"Sep."}
2792 MACRO {oct} {"Oct."}
2793 MACRO {nov} {"Nov."}
2794 MACRO {dec} {"Dec."}

```

## 2.8 Journal Abbreviations

### 2.8.1 Physics and astronomy

Borrowed from physjour.mbs of package custom-bib.

```

2795 MACRO {aa}{"Astron. \& Astrophys."}
2796 MACRO {aasup}{"Astron. \& Astrophys. Suppl. Ser."}
2797 MACRO {aj} {"Astron. J."}
2798 MACRO {aph} {"Acta Phys."}
2799 MACRO {advp} {"Adv. Phys."}
2800 MACRO {ajp} {"Amer. J. Phys."}
2801 MACRO {ajm} {"Amer. J. Math."}
2802 MACRO {amsci} {"Amer. Sci."}
2803 MACRO {anofd} {"Ann. Fluid Dyn."}
2804 MACRO {am} {"Ann. Math."}
2805 MACRO {ap} {"Ann. Phys. (NY)"}
2806 MACRO {adp} {"Ann. Phys. (Leipzig)"}
2807 MACRO {ao} {"Appl. Opt."}
2808 MACRO {apl} {"Appl. Phys. Lett."}
2809 MACRO {app} {"Astroparticle Phys."}
2810 MACRO {apj} {"Astrophys. J."}
2811 MACRO {apjsup} {"Astrophys. J. Suppl."}
2812 MACRO {apss} {"Astrophys. Space Sci."}
2813 MACRO {araa} {"Ann. Rev. Astron. Astrophys."}
2814 MACRO {baas} {"Bull. Amer. Astron. Soc."}
2815 MACRO {baps} {"Bull. Amer. Phys. Soc."}
2816 MACRO {cmp} {"Comm. Math. Phys."}
2817 MACRO {cpam} {"Commun. Pure Appl. Math."}
2818 MACRO {cppcf} {"Comm. Plasma Phys. \& Controlled Fusion"}
2819 MACRO {cpc} {"Comp. Phys. Comm."}
2820 MACRO {cvg} {"Class. Quant. Grav."}
2821 MACRO {cra} {"C. R. Acad. Sci. A"}
2822 MACRO {fed} {"Fusion Eng. \& Design"}
2823 MACRO {ft} {"Fusion Tech."}
2824 MACRO {grg} {"Gen. Relativ. Gravit."}
2825 MACRO {ieeens} {"IEEE Trans. Nucl. Sci."}
2826 MACRO {ieeeps} {"IEEE Trans. Plasma Sci."}
2827 MACRO {ijimw} {"Interntl. J. Infrared \& Millimeter Waves"}
2828 MACRO {ip} {"Infrared Phys."}
2829 MACRO {irp} {"Infrared Phys."}

```

2830 MACRO {jap} {"J. Appl. Phys."}  
 2831 MACRO {jasa} {"J. Acoust. Soc. America"}  
 2832 MACRO {jcp} {"J. Comp. Phys."}  
 2833 MACRO {jchp} {"J. Chem. Phys."}  
 2834 MACRO {jetp} {"Sov. Phys.--JETP"}  
 2835 MACRO {jfe} {"J. Fusion Energy"}  
 2836 MACRO {jfm} {"J. Fluid Mech."}  
 2837 MACRO {jmp} {"J. Math. Phys."}  
 2838 MACRO {jne} {"J. Nucl. Energy"}  
 2839 MACRO {jnec} {"J. Nucl. Energy, C: Plasma Phys., Accelerators, Thermonucl. Res."}  
 2840 MACRO {jnm} {"J. Nucl. Mat."}  
 2841 MACRO {jpc} {"J. Phys. Chem."}  
 2842 MACRO {jpp} {"J. Plasma Phys."}  
 2843 MACRO {jpsj} {"J. Phys. Soc. Japan"}  
 2844 MACRO {jsi} {"J. Sci. Instrum."}  
 2845 MACRO {jvst} {"J. Vac. Sci. & Tech."}  
 2846 MACRO {nat} {"Nature"}  
 2847 MACRO {nature} {"Nature"}  
 2848 MACRO {nedf} {"Nucl. Eng. & Design/Fusion"}  
 2849 MACRO {nf} {"Nucl. Fusion"}  
 2850 MACRO {nim} {"Nucl. Inst. & Meth."}  
 2851 MACRO {nimpr} {"Nucl. Inst. & Meth. in Phys. Res."}  
 2852 MACRO {np} {"Nucl. Phys."}  
 2853 MACRO {npb} {"Nucl. Phys. B"}  
 2854 MACRO {nt/f} {"Nucl. Tech./Fusion"}  
 2855 MACRO {npbpc} {"Nucl. Phys. B (Proc. Suppl.)"}  
 2856 MACRO {inc} {"Nuovo Cimento"}  
 2857 MACRO {nc} {"Nuovo Cimento"}  
 2858 MACRO {pf} {"Phys. Fluids"}  
 2859 MACRO {pfa} {"Phys. Fluids A: Fluid Dyn."}  
 2860 MACRO {pfb} {"Phys. Fluids B: Plasma Phys."}  
 2861 MACRO {pl} {"Phys. Lett."}  
 2862 MACRO {pla} {"Phys. Lett. A"}  
 2863 MACRO {plb} {"Phys. Lett. B"}  
 2864 MACRO {prep} {"Phys. Rep."}  
 2865 MACRO {pnas} {"Proc. Nat. Acad. Sci. USA"}  
 2866 MACRO {pp} {"Phys. Plasmas"}  
 2867 MACRO {pop} {"Phys. Plasmas"}  
 2868 MACRO {ppcf} {"Plasma Phys. & Controlled Fusion"}  
 2869 MACRO {phitrs1} {"Philos. Trans. Roy. Soc. London"}  
 2870 MACRO {prl} {"Phys. Rev. Lett."}  
 2871 MACRO {pr} {"Phys. Rev."}  
 2872 MACRO {physrev} {"Phys. Rev."}  
 2873 MACRO {pra} {"Phys. Rev. A"}  
 2874 MACRO {prb} {"Phys. Rev. B"}  
 2875 MACRO {prc} {"Phys. Rev. C"}  
 2876 MACRO {prd} {"Phys. Rev. D"}  
 2877 MACRO {pre} {"Phys. Rev. E"}  
 2878 MACRO {ps} {"Phys. Scripta"}  
 2879 MACRO {procrs1} {"Proc. Roy. Soc. London"}



2880 MACRO {rmp} {"Rev. Mod. Phys."}  
 2881 MACRO {rsi} {"Rev. Sci. Inst."}  
 2882 MACRO {science} {"Science"}  
 2883 MACRO {sciam} {"Sci. Am."}  
 2884 MACRO {sam} {"Stud. Appl. Math."}  
 2885 MACRO {st} {"Sky and Telesc."}

## 2.8.2 Supplementary Journal Names

Borrowed from suppjour.mbs of package custom-bib.

2886 MACRO {cjp} {"Czech. J. Phys."}  
 2887 MACRO {el} {"Europhys. Lett."}  
 2888 MACRO {en} {"Europhys. News"}  
 2889 MACRO {fujitsustj} {"FUJITSU Sci. Tech. J."}  
 2890 MACRO {ieeed} {"IEEE Trans. Electron Devices"}  
 2891 MACRO {ieeedim} {"IEEE Trans. Instrum. Meas."}  
 2892 MACRO {ieeedjqe} {"IEEE J. Quantum Electron."}  
 2893 MACRO {ieeedm} {"IEEE Trans. Magn."}  
 2894 MACRO {ieeedptl} {"IEEE Photonic Technol. Lett."}  
 2895 MACRO {ieeeduffc} {"IEEE Trans. Ultrason., Ferroelect., Freq. Cont."}  
 2896 MACRO {jem} {"J. Electron. Mater."}  
 2897 MACRO {jes} {"J. Electrochem. Soc."}  
 2898 MACRO {jetplett} {"JETP Lett."}  
 2899 MACRO {jjap} {"Japan. J. Appl. Phys."}  
 2900 MACRO {jpha} {"J. Phys. A: Math. Gen."}  
 2901 MACRO {jphb} {"J. Phys. B: At. Mol. Opt. Phys."}  
 2902 MACRO {jphbold} {"J. Phys. B: At. Mol. Phys."}  
 2903 MACRO {jphc} {"J. Phys.: Condens. Matter"}  
 2904 MACRO {jphcold} {"J. Phys. C: Solid State Phys."}  
 2905 MACRO {jphd} {"J. Phys. D: Appl. Phys."}  
 2906 MACRO {jvsta} {"J. Vac. Sci. Technol. A"}  
 2907 MACRO {jvstb} {"J. Vac. Sci. Technol. B"}  
 2908 MACRO {me} {"Microelectron. Eng."}  
 2909 MACRO {necrd} {"NEC Res.{&} Develop."}  
 2910 MACRO {pa} {"Physica A"}  
 2911 MACRO {pb} {"Physica B"}  
 2912 MACRO {pc} {"Physica C"}  
 2913 MACRO {pd} {"Physica D"}  
 2914 MACRO {procieed} {"Proc. IEEE"}  
 2915 MACRO {procspie} {"Proc. SPIE"}  
 2916 MACRO {pssa} {"Phys. Stat. Sol. A"}  
 2917 MACRO {pssb} {"Phys. Stat. Sol. B"}  
 2918 MACRO {rpp} {"Rep. Progr. Phys."}  
 2919 MACRO {sm} {"Synthet. Metal"}  
 2920 MACRO {sost} {"Solid State Technol."}  
 2921 MACRO {ss} {"Surf. Sci."}  
 2922 MACRO {ssc} {"Solid State Commun."}  
 2923 MACRO {sst} {"Semicond. Sci. Technol."}  
 2924 MACRO {suplatt} {"Superlatt. Microstr."}  
 2925 MACRO {sust} {"Supercond. Sci. Technol."}

2926 MACRO {znat} {"Z. Naturforsch."}

### 2.8.3 Optics

Borrowed from photjour.mbs.

2927 MACRO {appopt} {"Appl. Opt."}  
2928 MACRO {bell} {"Bell Syst. Tech. J."}  
2929 MACRO {ell} {"Electron. Lett."}  
2930 MACRO {jasp} {"J. Appl. Spectr."}  
2931 MACRO {jqe} {"IEEE J. Quantum Electron."}  
2932 MACRO {jlwt} {"J. Lightwave Technol."}  
2933 MACRO {jmo} {"J. Mod. Opt."}  
2934 MACRO {josa} {"J. Opt. Soc. America"}  
2935 MACRO {josaa} {"J. Opt. Soc. Amer.~A"}  
2936 MACRO {josab} {"J. Opt. Soc. Amer.~B"}  
2937 MACRO {jdp} {"J. Phys. (Paris)"}  
2938 MACRO {oc} {"Opt. Commun."}  
2939 MACRO {ol} {"Opt. Lett."}  
2940 MACRO {os} {"Opt. Spectrosc."}  
2941 MACRO {phtl} {"IEEE Photon. Technol. Lett."}  
2942 MACRO {pspie} {"Proc. Soc. Photo-Opt. Instrum. Eng."}  
2943 MACRO {vr} {"Vision Res."}  
2944 MACRO {zph} {"Z. f. Physik"}  
2945 MACRO {zphb} {"Z. f. Physik~B"}  
2946 MACRO {zphd} {"Z. f. Physik~D"}

### 2.8.4 Physics of condensed Matter

2947 MACRO {sse} {"Solid-State Electron."}  
2948 MACRO {pss} {"Phys. Sol. State"}  
2949 MACRO {nl} {"Nano Lett."}

### 2.8.5 Soviet and Russian journals

To be extended.

2950 MACRO {sjpp} {"Sov. J. Plasma Phys."}  
2951 MACRO {spd} {"Sov. Phys.--Doklady"}  
2952 MACRO {sptp} {"Sov. Phys.--Tech. Phys."}  
2953 MACRO {spu} {"Sov. Phys.--Uspekhi"}  
2954  $\langle\text{utf8}\rangle$ MACRO {ufn} {"\CYRU\CYRF\CYRN"}  
2955  $\langle\text{utf8}\rangle$ MACRO {ufn} {"VΦH"}  
2956 MACRO {pu} {"Phys.--Uspekhi"}  
2957 MACRO {sjot} {"Sov. J. Opt. Technol."}  
2958 MACRO {sjqe} {"Sov. J. Quantum Electron."}  
2959 MACRO {sleb} {"Sov. Phys.--Leb. Inst. Rep."}  
2960 MACRO {stph} {"Sov. Phys.--Techn. Phys."}  
2961 MACRO {stphl} {"Sov. Techn. Phys. Lett."}

## 2.9 Main cycle

```

2962
2963 READ
2964

```

## 2.10 Sorting

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

```

2965 ⟨*sort | natbib⟩
2966

```

sortify

```

2967 FUNCTION {sortify}
2968 { purify$
2969 ⟨!utf8⟩ "l" change.case$
2970 }
2971 ⟨/sort | natbib⟩
2972

```

sort.format.names

```

2973 ⟨*sort⟩
2974 %% =====
2975 %% This version from old Gost package
2976 %%<*/!natbib>
2977 FUNCTION {sort.format.names}
2978 { 's :=
2979   #1 'nameptr :=
2980   ""
2981   s num.names$ 'numnames :=
2982   numnames 'namesleft :=
2983     { namesleft #0 > }
2984     { nameptr #1 >
2985       { " " * }
2986       'skip$
2987       if$
2988       s nameptr
2989       "{vv{ } }{ll{ }}{ f{ }}{ jj{ }}"
2990       format.name$ 't :=
2991       nameptr numnames = t "others" = and
2992         { "et al" * }
2993         %{ bbl.etal * }
2994         { t sortify * }
2995       if$
2996       nameptr #1 + 'nameptr :=
2997       namesleft #1 - 'namesleft :=
2998     }
2999   while$
3000 }
3001 %%</!natbib>
3002 %% This version from plainnat.bst
3003 %% It ignores second and subsequent authors but include year.
3004 %%<*/!natbib>
3005 %FUNCTION {sort.format.names}

```

```

3006 %{ 's :=
3007 % #1 'nameptr :=
3008 % ""
3009 % s num.names$ 'numnames :=
3010 % numnames 'namesleft :=
3011 % { namesleft #0 > }
3012 % {
3013 %     s nameptr "{vv{ } }{ll{ }}{ ff{ }}{ jj{ }}" format.name$ 't :=
3014 %     nameptr #1 >
3015 %     {
3016 %         " " *
3017 %         namesleft #1 = t "others" = and
3018 %         { "zzzz" * }
3019 %         { numnames #2 > nameptr #2 = and
3020 %             { "zz" * year field.or.null * " " * }
3021 %             'skip$
3022 %             if$
3023 %             t sortify *
3024 %             }
3025 %             if$
3026 %             }
3027 %             { t sortify * }
3028 %             if$
3029 %             nameptr #1 + 'nameptr :=
3030 %             namesleft #1 - 'namesleft :=
3031 %             }
3032 % while$
3033 %}
3034 %%</natbib>
3035 %% =====
3036

```

sort.format.title

```

3037 FUNCTION {sort.format.title}
3038 { 't :=
3039   "A " #2
3040   "An " #3
3041   "The " #4 t chop.word % Removes "The " if any
3042   chop.word % Removes "An " if any
3043   chop.word % Removes "A " if any
3044   sortify
3045   #1 global.max$ substring$
3046 }
3047

```

author.sort

```

3048 %% =====
3049 %% This version from old gost package.
3050 %%
3051 <!*natbib>
3052 FUNCTION {author.sort}

```

```

3053 { author empty$
3054   { key empty$
3055     { "to sort, need author or key in " cite$ * warning$
3056       ""
3057     }
3058     { key sortify }
3059   if$
3060 }
3061 {
3062   author num.names$ #4 <
3063   {author sort.format.names }
3064   {title sort.format.title}
3065   if$
3066 }
3067 if$
3068 }
3069 <\/!natbib>
3070 %% This version from plainnat.bst
3071 < *natbib>
3072 FUNCTION {author.sort}
3073 { author empty$
3074   { key empty$
3075     { "to sort, need author or key in " cite$ * warning$
3076       ""
3077     }
3078     { key sortify }
3079   if$
3080 }
3081 { author sort.format.names }
3082 if$
3083 }
3084 <\/natbib>
3085 %% =====
3086

```

author.title.sort

```

3087 < *!natbib | natbib>
3088 FUNCTION {author.title.sort}
3089 { author empty$
3090   { title empty$
3091     { key empty$
3092       { "to sort, need author, title, or key in " cite$ * warning$
3093         ""
3094       }
3095       { key sortify }
3096     if$
3097   }
3098   { title sort.format.title }
3099   if$
3100 }
3101 {

```

```

3102     author num.names$ #4 <
3103     {author sort.format.names }
3104     {title sort.format.title}
3105     if$
3106   }
3107   if$
3108 }
3109 <\/!natbib | natbib>
3110 < *natbib | natbib>
3111 FUNCTION {author.editor.sort}
3112 { author empty$
3113   { editor empty$
3114     { key empty$
3115       { "to sort, need author, editor, or key in " cite$ * warning$
3116         ""
3117       }
3118       { key sortify }
3119     if$
3120   }
3121   { editor sort.format.names }
3122   if$
3123 }
3124 { author sort.format.names }
3125 if$
3126 }
3127
3128 FUNCTION {author.organization.sort}
3129 { author empty$
3130   { organization empty$
3131     { key empty$
3132       { "to sort, need author, organization, or key in " cite$ * warning$
3133         ""
3134       }
3135       { key sortify }
3136     if$
3137   }
3138   { "The " #4 organization chop.word sortify }
3139   if$
3140 }
3141 { author sort.format.names }
3142 if$
3143 }
3144
3145 FUNCTION {editor.organization.sort}
3146 { editor empty$
3147   { organization empty$
3148     { key empty$
3149       { "to sort, need editor, organization, or key in " cite$ * warning$
3150         ""
3151       }

```

```

3152         { key sortify }
3153     if$
3154 }
3155 { "The " #4 organization chop.word sortify }
3156 if$
3157 }
3158 { editor sort.format.names }
3159 if$
3160 }
3161 </natbib|natbib>
3162
3163

```

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

```

3164 <!*natbib>
3165 FUNCTION {presort}%#1
3166 {
3167   author.title.sort
3168   "   "
3169   *
3170   year field.or.null sortify
3171   *
3172   "   "
3173   *
3174   title field.or.null
3175   sort.format.title
3176   *
3177   #1 entry.max$ substring$
3178   'sort.key$ :=
3179 }
3180 </!natbib>
3181 <*natbib>
3182 FUNCTION {presort}%#2
3183 { calc.label
3184   label sortify
3185   %author.title.sort
3186   "   "
3187   *
3188   % ===== plainnat.bst =====
3189   % type$ "book" =
3190   % type$ "inbook" =
3191   % or
3192   % 'author.editor.sort
3193   % { type$ "proceedings" =
3194   %   'editor.organization.sort
3195   %   { type$ "manual" =
3196   %     'author.organization.sort
3197   %     'author.sort
3198   %   if$
3199   % }
3200   % if$

```

```

3201 %    }
3202 %  if$
3203   author.title.sort
3204   "      "
3205   *
3206   year field.or.null sortify
3207   *
3208   "      "
3209   *
3210   %cite$
3211   title field.or.null sort.format.title
3212   *
3213   #1 entry.max$ substring$
3214   'sort.label :=
3215   sort.label *
3216   % =====
3217   #1 entry.max$ substring$
3218   'sort.key$ :=
3219 }
3220 </natbib>
3221 </sort>
3222
3223 <!*sort>
3224 <*natbib>
3225 INTEGERS { seq.num }
3226
3227 FUNCTION {init.seq}
3228 { #0 'seq.num :=}
3229
3230 EXECUTE {init.seq}
3231
3232 FUNCTION {int.to.fix}
3233 { "000000000" swap$ int.to.str$ *
3234   #-1 #10 substring$
3235 }
3236
3237 FUNCTION {presort}%#3
3238 {
3239   calc.label           % computes label
3240   label sortify        % initiates sort.label
3241   "      "
3242   *
3243   seq.num #1 + 'seq.num := % advance seq.num
3244   seq.num int.to.fix      % prepend seq.num with 0s
3245   'sort.label :=          % set sort.label to seq.num
3246   sort.label *            % append seq.num to label
3247   #1 entry.max$ substring$ % cut if too long
3248   'sort.key$ :=          % set sort.key$
3249 }
3250 </natbib>

```



```

3251 </!sort>
3252
3253 < *sort | natbib>
3254 ITERATE {presort}
3255
3256 SORT
3257
3258 </sort | natbib>
3259

```

## 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.

```

3260 <!natbib>STRINGS { longest.label }
3261 <natbib>STRINGS { longest.label last.label next.extra }
3262
3263 <!natbib>INTEGERS { number.label longest.label.width }
3264 <natbib>INTEGERS { number.label longest.label.width last.extra.num }
3265

```

`ize.longest.label` Initialize those string.

```

3266 < *!natbib>
3267 FUNCTION {initialize.longest.label}
3268 { "" 'longest.label :=
3269   #1 'number.label :=
3270   #0 'longest.label.width :=
3271 }
3272 </!natbib>
3273 < *natbib>
3274 FUNCTION {initialize.longest.label}
3275 { "" 'longest.label :=
3276   #0 int.to.chr$ 'last.label :=
3277   "" 'next.extra :=
3278   #0 'longest.label.width :=
3279   #0 'last.extra.num :=
3280   #0 'number.label :=
3281 }
3282 </natbib>
3283
3284 EXECUTE {initialize.longest.label}
3285

```

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

```

3286 < *!natbib>
3287 FUNCTION {forward.pass}
3288 { number.label int.to.str$ 'label :=
3289   number.label #1 + 'number.label :=

```

```

3290   label width$ longest.label.width >
3291     { label 'longest.label :=
3292       label width$ 'longest.label.width :=
3293     }
3294     'skip$
3295   if$
3296 }
3297 </!natbib>
3298 <natbib>
3299 FUNCTION {forward.pass}
3300 { last.label label =
3301   { last.extra.num #1 + 'last.extra.num :=
3302     last.extra.num int.to.chr$ 'extra.label :=
3303   }
3304   { "a" chr.to.int$ 'last.extra.num :=
3305     "" 'extra.label :=
3306     label 'last.label :=
3307   }
3308   if$
3309   number.label #1 + 'number.label :=
3310 }
3311 </natbib>
3312
3313 ITERATE {forward.pass}
3314

```

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

```

3315 <natbib>
3316 FUNCTION {reverse.pass}
3317 { next.extra "b" =
3318   { "a" 'extra.label := }
3319   'skip$
3320   if$
3321   extra.label 'next.extra :=
3322   extra.label
3323   duplicate$ empty$
3324   'skip$
3325   { "{\natexlab{" swap$ * "}}" * }
3326   if$
3327   'extra.label :=
3328   label extra.label * 'label :=
3329 }
3330
3331 REVERSE {reverse.pass}
3332
3333 FUNCTION {bib.sort.order}
3334 { sort.label 'sort.key$ :=
3335 }
3336
3337 ITERATE {bib.sort.order}
3338

```

```

3339 SORT
3340  $\langle$ /natbib $\rangle$ 
3341

```

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

```

3342 FUNCTION {begin.bib}
3343 { "\begin{thebibliography}{\longest.label *}" * write$ newline$
3344   "\def\selectlanguageifdefined#1{" write$ newline$
3345   "\expandafter\ifx\csname date#1\endcsname\relax" write$ newline$
3346   % "\else\language\csname l@#1\endcsname\fi" write$ newline$
3347   "\else\selectlanguage{#1}\fi" write$ newline$
3348   "\providecommand*\href[2]{\small #2}" write$ newline$
3349   "\providecommand*\url[1]{\small #1}" write$ newline$
3350   "\providecommand*\BibUrl[1]{\url{#1}}" write$ newline$
3351   "\providecommand*\BibAnnote[1]{}" write$ newline$
3352   "\providecommand*\BibEmph[1]{#1}" write$ newline$
3353  $\langle$ *modern!modern $\rangle$ 
3354   %"\ProvideTextCommandDefault{\cyrdash}{---}" write$ newline$
3355   %\DeclareUTFcharacter[\UTFencname]{x2014}{\cyrdash}
3356   %"\let\cyrdash\textemdash" write$ newline$
3357   %"\ProvideTextCommandDefault{\cyrdash}{\hbox to.8em{--\hss--}}" write$ newline$
3358   %"\ProvideTextCommandDefault{\cyrdash}{\textemdash}" write$ newline$
3359   "\ProvideTextCommandDefault{\cyrdash}{\iflanguage{russian}{\hbox to.8em{--\hss--}}{\textemdash}}" write$ n
3360   %%"\ProvideTextCommandDefault{\cyrdash}{%" write$ newline$
3361   %%" \iflanguage{russian}{\hbox to.8em{--\hss--}}{%" write$ newline$
3362   %%" \iflanguage{ukrainian}{\hbox to.8em{--\hss--}}{\textemdash}}" write$ newline$
3363   "\providecommand*\BibDash{\ifdim\lastskip>0pt\unskip\nobreak\hskip.2em plus 0.1em\fi}" write$ newline$
3364   "\cyrdash\hskip.2em plus 0.1em\ignorespaces" write$ newline$
3365   "\renewcommand{\newblock}{\ignorespaces}" write$ newline$
3366  $\langle$ /modern!modern $\rangle$ 
3367  $\langle$ natbib $\rangle$  "\providecommand{\natexlab}[1]{#1}" write$ newline$
3368   preamble$ empty$
3369   'skip$
3370   { preamble$ write$ newline$ }
3371   if$
3372 }
3373
3374
3375 EXECUTE {begin.bib}
3376
3377 EXECUTE {init.state.consts}
3378
3379 ITERATE {call.type$}
3380

```

end.bib

```

3381 FUNCTION {end.bib}
3382 { newline$
3383   % "\catcode'\=11" write$ newline$
3384   "\end{thebibliography}" write$ newline$

```

```

3385 }
3386
3387 EXECUTE {end.bib}
3388
3389 </bst>

```

That's all, Folks!

## Change History

v0.8		Options modern, long, eprint . . . . . 1
General: \BibAnnote added . . . . . 1		Refactoring, Documentation . . . . . 1
\BibEmph added . . . . . 1		Strict option . . . . . 1
\BibUrl added . . . . . 1		Support for natbib package . . . . . 1
Entry ANNOTE added . . . . . 1		Thesis entry, report entry . . . . . 1
v0.9		
General: Bug fix in INPROCEEDINGS . . . 1	v1.2a	General: Default for \cyrdash added . . . . 1
Bug fix in names and date formatting . . 1	v1.2b	
v1.0		General: numpages renamed to pagetotal . . 1
General: Bug fix (long annote) . . . . . 1	v1.2c	
v1.1		General: eid field added . . . . . 1
General: Added German, French, Italian		fix Gost2003: — replaced by \BibDash . . . 1
languages . . . . . 1		langid field added . . . . . 1
Entry ONLINE . . . . . 1	v1.2d	
Gost705.dtx borrowed from Disser pkg . . 1	General: \bbljan e.t.c. macros removed . . . 1	
Upload to CTAN . . . . . 1	v1.2e	
v1.2		General: bbl.url added to replace URL string 1
General: Entries eprint, eprintclass,	v1.2f	
eprinttype . . . . . 1	General: patent entry added . . . . . 1	
Fix bbl.urldate for ukrainian (Andrey	v1.2g	
Shvajkoy) . . . . . 1	General: minor changes in documentation . . 1	
Medium field . . . . . 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		
\& . . . . . 2795, 2796, 2818,	add.doi . . . . . 50	author.key.label . . . . . 29
2822, 2827, 2845, 2848,	add.media . . . . . 50	author.sort . . . . . 76
2850, 2851, 2868, 2909	add.number . . . . . 52	author.title.sort . . . . . 77
\' . . . . . 406	and . . . . . 12	bbl.and . . . . . 18
\, . . . . 718, 720, 727, 735, 742	article . . . . . 54	bbl.chief . . . . . 24
\. . . . . 1191	author.after . . . . . 47	bbl.cmplr . . . . . 15
\/ . . . . . 3383	author.before . . . . . 46	bbl.dscithesis . . . . . 23
add.blank . . . . . 12	author.editor.full . . . . . 36	bbl.edby . . . . . 15
	author.full . . . . . 36	bbl.edition . . . . . 15

bbl.etal	18	end.bib	83	inproceedings	58
bbl.executor	25	eng.ord	41	internet	63
bbl.iin	20	extract.num	40	is.num	40
bbl.iissue	17	field.or.null	14	make.full.names	36
bbl.iiss	17	fin.entry	10	manual	59
bbl.in	20	format.address.publisher.date	34	mastersthesis	67
bbl.media.elres	24	format.address.publisher	34	misc	61
bbl.media.text	24	format.annotate	49	multi.page.check	42
bbl.media	25	format.authors.after	32	n.dashify	38
bbl.mthesis	22	format.authors	29	new.block.checka	13
bbl.nnoaddress	23	format.bookauthors.after	32	new.block.checkb	13
bbl.nnopublisher	24	format.bookauthors	31	new.block	11
bbl.nnr	19	format.btitle	39	new.colon	11
bbl.nnumber	18	format.bvolume	40	new.dblslash.checka	14
bbl.nopublisher	23	format.chapter.pages	45	new.dblslash	11
bbl.nr	19	format.chief.after	32	new.semicolon	11
bbl.number	19	format.compiler.after	32	new.sentence.checka	13
bbl.of	17	format.date	34	new.sentence.checkb	13
bbl.pages	20	format.edition	42	new.sentence	11
bbl.page	20	format.editors.after	32	new.slash	11
bbl.phdthesis	23	format.eprint	51	non.stop	13
bbl.ppages	21	format.executor.after	32	not	12
bbl.ppage	21	format.full.names	36	online	62
bbl.priority	26	format.isbn	49	or	12
bbl.publ	26	format.key	29	output.address.publisher.date	35
bbl.req	25	format.lab.names	28	output.bibitem	35, 36
bbl.techreport	22	format.month	33	output.check	10
bbl.urldate	22	format.names.rev	28	output.nonnull	9
bbl.url	21	format.names	27	output.url	49
bbl.vvolume	16	format.number.series	40	output	10
bbl.vvol	16	format.number	45	patent	60
begin.bib	83	format.pages.page	43	phdthesis	66
bookauthor.after	48	format.pages	43	presort	79
bookauthor.before	47	format.prioritydate	53	proceedings	57
booklet	55	format.publicationdate	53	report	64
book	54	format.requestdate	53	reverse.pass	82
bracify	14	format.techrep.type.number	46	sort.format.names	75
change.language	9	format.thesis.type	45	sort.format.title	76
chop.word	12	format.title	33	sortify	75
conference	69	format.type.number	53	space.word	15
convert.edition	41	format.url	49	techreport	69
default.type	70	format.vol.num.pages	44	thesis	63
dscithesis	68	format.volume	44	tie.connect	39
editor.full	36	ielectronic	63	tie.or.space.connect	39
editor.organization.after	48	inbook	56	unpublished	62
either.or.chec	39	incollection	57	webpage	63
emphasize	14	init.state.consts	9	word.in	39
empty.misc.check	45	initialize.longest.label	81	www	63
enclose.round.brackets	14			\‘	726
enclose.square.brackets	14				

$\sqcup$ . . . . .	360, 362, 365, 369, 659, 662, 675, 678, 708, 722, 723, 724, 739, 740, 741, 749, 751, 752, 753, 759, 761, 762, 763, 769, 771, 772, 773, 791, 794, 1132, 1139, 1153	$\backslash$ cyn . . . . .	1194	$\backslash$ ifx . . . . .	3345
		$\backslash$ CYRA . . . . .	1179, 1191	$\backslash$ ignorespaces . . . .	3364, 3365
		$\backslash$ CYRB . . . . .	751, 771		
		$\backslash$ cyrch . . . . .	694	<b>L</b>	
		$\backslash$ cyrdash . . . . .	3354, 3355, 3356, 3357, 3358, 3359, 3360, 3364	$\backslash$ language . . . . .	3346
		$\backslash$ CYRE . . . . .	794	$\backslash$ lastskip . . . . .	3363
		$\backslash$ CYREREV . . . . .	791	$\backslash$ ldots . . . . .	722, 724, 739, 741
		$\backslash$ CYRF . . . . .	1173, 2954	$\backslash$ let . . . . .	3356
<b>B</b>		$\backslash$ cyrf . . . . .	707	<b>N</b>	
$\backslash$ bbland . . . . .	510	$\backslash$ CYRI . . . . .	1185, 1188	$\backslash$ natexlab . . . . .	3325, 3367
$\backslash$ bbledby . . . . .	358	$\backslash$ cyrishrt . . . .	791, 794, 1182	$\backslash$ newblock . . . . .	140, 3365
$\backslash$ bbledition . . . . .	392	$\backslash$ CYRO . . . . .	694, 1197	$\backslash$ nobreak . . . . .	3363
$\backslash$ bblin . . . . .	583	$\backslash$ CYRR . . . . .	659, 662		
$\backslash$ bbllss . . . . .	458	$\backslash$ cyrsftsn . . . .	822, 1185, 1188	<b>P</b>	
$\backslash$ bbllssue . . . . .	441	$\backslash$ CYRYA . . . . .	1170	$\backslash$ providecommand . . . . .	
$\backslash$ bblno . . . . .	554, 570	$\backslash$ cyryu . . . . .	1185, 1188	. . . . .	3348, 3349, 3350, 3351, 3352, 3363, 3367
$\backslash$ bbllNumber . . . . .	528	$\backslash$ cyrgyz . . . . .	659, 662	$\backslash$ ProvideTextCommandDefault	
$\backslash$ bbllnumber . . . . .	541			. . . . .	3354, 3357, 3358, 3359, 3360
$\backslash$ bblof . . . . .	475, 492	<b>D</b>			
$\backslash$ bbLP . . . . .	642	$\backslash$ DeclareUTFcharacter . .	3355	<b>R</b>	
$\backslash$ bbLP . . . . .	616	$\backslash$ def . . . . .	3344	$\backslash$ relax . . . . .	3345
$\backslash$ bbLPp . . . . .	629			$\backslash$ renewcommand . . . . .	3365
$\backslash$ bbLPp . . . . .	603	<b>E</b>			
$\backslash$ bbLVol . . . . .	428	$\backslash$ else . . . . .	3346, 3347	<b>S</b>	
$\backslash$ bbLVolume . . . . .	415	$\backslash$ end . . . . .	3384	$\backslash$ selectlanguage . . . . .	3347
$\backslash$ begin . . . . .	3343	$\backslash$ endcsname . . . . .	3345, 3346	$\backslash$ selectlanguageifdefined	
$\backslash$ BibAnnote . . . . .	1917, 3351	$\backslash$ expandafter . . . . .	3345	. . . . .	124, 1291, 1403, 1425, 3344
$\backslash$ BibDash . . . . .	138, 1222, 3363	<b>F</b>		$\backslash$ small . . . . .	3348, 3349
$\backslash$ BibEmph . . . . .	328, 3352	$\backslash$ fi . . . . .	3346, 3347, 3363		
$\backslash$ bibitem . . . . .	11, 12, 13, 1280, 1382, 1411	<b>H</b>		<b>T</b>	
$\backslash$ BibUrl . . . . .	1895, 1896, 3350	$\backslash$ hbox . . . . .	3357, 3359, 3361, 3362	$\backslash$ textemdash . . . . .	
		$\backslash$ href . . . . .	1926, 1938, 1971, 1973, 1976, 1999, 2016, 3348	. . . . .	3356, 3358, 3359, 3362
<b>C</b>		$\backslash$ hskip . . . . .	3363, 3364	$\backslash$ textnumero . . . . .	560, 574
$\backslash$ catcode . . . . .	3383	$\backslash$ hss . . . . .	3357, 3359, 3361, 3362		
$\backslash$ cite . . . . .	21, 24, 25, 26, 27, 28			<b>U</b>	
$\backslash$ citeauthor . . . . .	29, 30	<b>I</b>		$\backslash$ unskip . . . . .	3363
$\backslash$ citete . . . . .	22, 23	$\backslash$ ifdim . . . . .	3363	$\backslash$ url . . . . .	3349, 3350
$\backslash$ citeyear . . . . .	31	$\backslash$ iflanguage . . . .	3359, 3361, 3362	$\backslash$ UTFencname . . . . .	3355
$\backslash$ csname . . . . .	3345, 3346				