The GOST package

Igor Kotelnikov, Leonid Sinev, *
https://github.com/kia999/GOST/

Version ? released ?

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.

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Kotelnikov to the present status and some code was borrowed from disser package developed by Stanislav Kruchinin and unpublished work by Artem Petrenkov.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 7.80 -2000 Bibliographic record. Heading. General requirements and rules.

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

GOST 7.1 -2003 Bibliographic record. Bibliographic description. 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 7.0.5-2008 Bibliographic reference. General requirements and rules of making.

Etc.

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) versions of each BibTeX style, in each case offering a choice of sorted and unsorted.

*With small temporary fix by Oleksandr Baranovskyi.

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	eprint	long	sort	natbib
gost2003		+	+			
gost2003s		+	+		+	
gost2008			+			
gost2008n			+			+
gost2008l			+	+		
gost2008s			+		+	
gost2008ns			+		+	+
gost2008ls			+	+	+	
ugost2003	+	+	+			
ugost2003s	+	+	+		+	
ugost2008	+		+			
ugost2008n	+		+			+
ugost2008l	+		+	+		
ugost2008s	+		+		+	
ugost2008ns	+		+		+	+
ugost2008ls	+		+	+	+	
Style	utf8	strict	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.

The `strict` option provides conformance to the Standard 7.1-2003. The bibstyles 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 bibstyles compiled without `strict` option meets the Standard 7.0.5-2008 which can be thought off as a relaxed version of the Standards 7.1-2003. These bibstyles bear the name `gost2008` with possible suffixes `s`, `l`, `n`.

If the number of authors exceeds 4, modern styles cut the list of authors to at most 4 persons as prescibed by the Standards. Option `long` overrides this rule to provide backward compatibility with the package disser by Stanislab 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 the 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 that 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

2 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 `langid="ukrainian"` or `langid="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` and `ugost*.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 `bibtex.exe` or `bibtex8.exe` 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.

3 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 `russian`. It prints a so called Cyrillic dash (`\cyrdash`), which is 20% shorter then ordinary LaTeX dash (`---`), and puts unbreakable space before `\cyrdash` so that dash never appears in the beginning of a line.

4 Where to get

1. http://ctan.org/pkg/gost.

2. http://github.com/kia999/gost.

5 Version history

Version 1.2l (2021.02.05)

Bug fixed: removed `\endofline`.

Version 1.2k (2020.12.29)

1. All stuff is now generated from `gost.dtx`.
2. Limited support of the `date` field added.
3. Formatting of `doi` field updated: `http://dx.doi.org` changed to `https://doi.org`.
4. Restricted support of `date` field added.
5. Documentation and examples update (thanks to Leonid Sinev).

Version 1.2i (2017.01.12)

1. Documentation and examples update (thanks to Leonid Sinev).
2. Restored `@MastersThesis` instead of `@MasterThesis` (thanks to Leonid Sinev).
3. `media="eresource"` is introduced in addition to `media="online"` and `media="text"`; if present, the `media` field is not ignored any more in modern bst-styles compiled without the `strict` option.
4. `location` field is introduced as an alias of `address` field.
5. `@DSCISTHESIS` entry renamed to `@DOCTHESIS`.
6. `school` field in `@THESIS` and similar entries is replaced by `institution` to comply with `biblatex-gost` style.

Version 1.2h (2016.08.21)

1. Minor changes in documentation.

Version 1.2g (2016.07.25)

1. Minor changes in documentation.

Version 1.2f (2016.07.12)

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

Version 1.2e (2016.07.07)

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

Version 1.2d (2015.02.18)

1. jan, feb, etc. macros fixed.
2. New macro `format.month`.

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 (`\cyrdash`) only for `russian` 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.)

Version 1.2a (2012.08.31)

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

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.

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.

5.1 Older versions

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.

6 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}`.

```
⟨*bst⟩
%% This bibstyle attempts to format bibliography according to
⟨strict⟩%% GOST 7.1-2003 for bibliographic records.
⟨!strict⟩%% GOST 7.0.5-2008 for bibliographic reference.
⟨*natbib⟩%%
%%------------------------------------------------------------------
%% This is an author-year citation style bibliography.
%% It requires a special package file to function properly
%% such as natbib.sty by Patrick W. Daly.
%% The form of the \bibitem entries is
%%   \bibitem[Jones et al.(1990)]{key}...
%%   \bibitem[Jones et al.(1990)Jones, Baker, and Smith]{key}...
%% where the label part [in brackets] consists of the author names,
%% as they should appear in the citation, with the year in parentheses following.
%% There must be no space before the opening parenthesis!
%% A full list of authors may also follow the year.
%% In natbib.sty, it is possible to define the type of enclosures that is
%% really wanted (brackets or parentheses), but in either case, there must
%% be parentheses in the label.
%% The \cite command functions as follows:
%%   \citet{key}              => Jones et al. (1990)
%%   \citet*{key}             => Jones, Baker, and Smith (1990)
%%   \cite{key}               => (Jones et al., 1990)
%%   \cite*{key}              => (Jones, Baker, and Smith, 1990)
%%   \cite[chap. 2]{key}      => (Jones et al., 1990, chap. 2)
%%   \cite[e.g.][]{key}       => (e.g. Jones et al., 1990)
%%   \cite[e.g.][p. 32]{key}  => (e.g. Jones et al., p. 32)
%%   \citeauthor{key}         => Jones et al.
%%   \citeauthor*{key}        => Jones, Baker, and Smith
%%   \citeyear{key}           => 1990
%%------------------------------------------------------------------
⟨/natbib⟩

```

6.1 Fields

Enlist all entry fields allowed in a bibliographic database. Most fields are common for many standard `bst` styles. Nonlisted fields are just ignored by BibTeX.

```
ENTRY
  { address
    annote
    author
    booktitle
    bookauthor
    chapter
    edition
    editor
    compiler
    howpublished
    institution
    journal
    key
    %major          % new in v.1.2i, alias for speciality, not implemented
    majorcode       % new in v.1.2i, alias for specialitycode
    month
    note
    number
    organization
    pages
    eid             % new in v1.2c
    publisher
    school          % alias for institution
    series
    %speciality     % new in v.1.2i, eqv. to major in biblatex-gost, not implemented yet...
    specialitycode  % new in v.1.2i, alias of number, eqv. to majorcode in biblatex-gost
    title
    %medium         % new in v1.2; renamed to media.
    media           % new in v1.2f
    type
    volume
    year
    language
    langid          % new in v1.2c
    booklanguage
```

Entries borrowed from biblatex.

```
    date            % new in v1.2i
    pagetotal
    url
    urldate
    isbn
    doi
    eprint
    eprinttype      % = archivePrefix
    eprintclass     % = primaryClass
```

```
    % new in v1.2f:
    % appear in biblatex:
    %addendum       % not implemented yet...
    holder          % see patent
    location        % new in v.1.2i, alias of address
    %subtitle       % not implemented yet...
    titleaddon      % new in v.1.2i, see @thesis
    %version        % not implemented yet...
    % Appear in biblatex-gost for @patent entry:
    authorcountry   % country of the patent authors
    credits         % statement of responsibility, other than provided in Biblatex
    ipc             % Code of the International Patent Classification
    %media          % General material designation NOTE: medium in the above
    requestnumber   % Registration number of the application to the patent document
    publicationdate % Date of publication
    publication     %  and information on the official gazette, which published patent
    prioritydate    % Information about the convention priority: the date of filing of the application,
    prioritynumber  %   number and
    prioritycountry %   country name of convention priority.
    requestdate     % ??
  }
  {}
⟨!natbib⟩  { label }
⟨natbib⟩  { label extra.label sort.label short.list }

```

6.2 Output functions

Declare internal variables and constants used for formatting bibliographic records. Other variables are defined below when needed.

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

STRINGS { curlanguage }

STRINGS { s t }

STRINGS { y m d }   % new in v.1.2j

```

`init.state.consts` Set constants that designate various output states which are kept by ‘output.state‘ integer. The

latter is checked and updated by a set of functions such as ‘output‘ to be defined below.

```
FUNCTION {init.state.consts}
{ #0 'before.all :=
  #1 'mid.sentence :=
  #2 'after.sentence :=
  #3 'after.block :=
  #4 'after.dblslash :=
  #5 'after.slash :=
  #6 'after.colon :=
  #7 'after.semicolon :=
}

```

`set.language` Sets current language `curlanguage`. Called by `bibitem.output` before any other function.

```
FUNCTION {set.language}
{ langid empty$
    { language empty$
        { "english" 'curlanguage := }
        { language  'curlanguage := }
      if$
    }
    { langid  'curlanguage := }
  if$
}

```

`reset.language` Reset current language to `booklanguage` if provided. Called by `output.nonnull` after double slash so that the rest of the record is formatted with `booklanguage`.

```
FUNCTION {reset.language}
{ booklanguage empty$
    { "" }
    { booklanguage  'curlanguage :=
      "\selectlanguageifdefined{"
      curlanguage *
      "}" *
    }
  if$
}

```

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

`output.nonnull` Writes the last literal in the stack to output buffer assuming that it is not empty and adds an appropriate punctuation symbol.

```
FUNCTION {output.nonnull}
{
  swap$
  output.state mid.sentence =
    { ", " * write$ }
    { output.state after.block =
        { add.period$ write$
          " \BibDash " write$
```

```
          newline$
          "\newblock " write$
        }
        { output.state before.all =
            'write$
            { output.state after.dblslash =
                { "~//" * reset.language * " " * write$ }
                { output.state after.slash =
                    { "~/ " * write$ }
                    { output.state after.colon =
                        { "~: " * write$ }
                        { output.state after.semicolon =
                            { "~; " * write$ }
                        { add.period$ " " * write$ }
                       if$
                        }
                      if$
                    }
                  if$
                }
              if$
            }
          if$
        }
      if$
      mid.sentence 'output.state :=
    }
  if$
}

```

`output` Calls `output.nonnull` if the last literal string in the stack is not empty; otherwise it discards the literal.

```
FUNCTION {output}
{ duplicate$ empty$
    'pop$
    'output.nonnull
  if$
}

```

`output.check` Does the same but also warns if the indicated field is empty. Needs two literals in the stack: the field and the name of the field, e.g., `author "author" output.check`.

```
FUNCTION {output.check}
{ 't :=
  duplicate$ empty$
    { pop$
      "empty " t * " in " * cite$ * warning$
    }
    'output.nonnull
  if$
```

```
}
```

`fin.entry` `fin.entry` finalizes current entry. It writes dot, if no dot is found in stack, and starts new line.

```
FUNCTION {fin.entry}
{ add.period$
  write$
  newline$
}
```

Declare a family of functions to put punctuation marks depending on the current status of the output stack.

`new.block` This just checks output state and revert it to another state if required. Checking output state prevents occasional doubling of punctuation marks.

```
FUNCTION {new.block}
{ output.state before.all =
    'skip$
    { after.block 'output.state := }
  if$
}
```

`new.dblslash`

```
FUNCTION {new.dblslash}
{ output.state before.all =
    'skip$
    { after.dblslash 'output.state := }
  if$
}
```

`new.slash`

```
FUNCTION {new.slash}
{ output.state before.all =
    'skip$
    { after.slash 'output.state := }
  if$
}
```

`new.colon`

```
FUNCTION {new.colon}
{ output.state before.all =
    'skip$
    { after.colon 'output.state := }
  if$
}
```

`new.semicolon`

```
FUNCTION {new.semicolon}
```

```
{ output.state before.all =
    'skip$
    { after.semicolon 'output.state := }
  if$
}

new.sentence

FUNCTION {new.sentence}
{ output.state after.block =
    'skip$
    { output.state before.all =
        'skip$
        { after.sentence 'output.state := }
      if$
    }
  if$
}

   add.blank

FUNCTION {add.blank}
{ " " * before.all 'output.state :=
}
```

6.3 Logical functions and various checks

Declare few logical functions.

```
       not

FUNCTION {not}
{   { #0 }
    { #1 }
  if$
}

       and

FUNCTION {and}
{   'skip$
    { pop$ #0 }
  if$
}

        or

FUNCTION {or}
{   { pop$ #1 }
    'skip$
  if$
}
```

non.stop ⟨NB: What's the hell? Never used.⟩

```
%FUNCTION {non.stop}
%{ duplicate$
%   "}" * add.period$
%   #-1 #1 substring$ "." =
%}
%
```

new.block.checka Adds `new.block` if the last literal in stack is not empty.

```
FUNCTION {new.block.checka}
{ empty$
    'skip$
    'new.block
  if$
}

```

new.block.checkb Adds `new.block` if either of the two last literals in the stack is not empty.

```
FUNCTION {new.block.checkb}
{ empty$
  swap$ empty$
  and
    'skip$
    'new.block
  if$
}

```

w.sentence.checka Adds `new.sentence` if the last literal in stack is not empty.

```
FUNCTION {new.sentence.checka}
{ empty$
    'skip$
    'new.sentence
  if$
}

```

w.sentence.checkb Adds `new.sentence` if either of the two last literals in the stack is not empty.

```
FUNCTION {new.sentence.checkb}
{ empty$
  swap$ empty$
  and
    'skip$
    'new.sentence
  if$
}

```

w.dblslash.checka For `online` entry.

```
FUNCTION {new.dblslash.checka}
{ empty$
    'skip$
```

```
    'new.dblslash
  if$
}

```

`field.or.null` Replaces an empty field with null string `""`.

```
FUNCTION {field.or.null}
{ duplicate$ empty$
    { pop$ "" }
    'skip$
  if$
}

```

`either.or.check` ⟨NB: Move upwards⟩

```
FUNCTION {either.or.check}
{ empty$
    'pop$
    { "can't use both " swap$ * " fields in " * cite$ * warning$ }
  if$
}

```

6.4 String Functions

`spaces.around` Inserts a space before and after last literal in the stack.

```
FUNCTION {spaces.around}
{ " " swap$ * " " * }

```

`emphasize` Emphasizes the last literal in the stack if it is not empty. v1.2k: `emphasize` now returns empty field if last literal in the stack was empty.

```
FUNCTION {emphasize}
{ duplicate$ empty$
    %{ pop$ "" }
    'skip$       % v.1.2k
    { "\BibEmph{" swap$ * "}" * }
  if$
}

```

`bracify` New in v.1.2. An idea borrowed from apsrev4-1.bst. Encloses last literal in the stack by braces even if it is empty. Note that braces are normally not printed by LaTeX.

```
FUNCTION {bracify}
{ duplicate$ empty$
    { pop$ "{}" }
    { "{" swap$ * "}" * }
  if$
}

```

`bracketise` This and the next functions are used to enclose last word by square and round brackets. In contrast to `bracify` function they push null string if the last literal is empty.

```
FUNCTION {bracketise}
{
  duplicate$ empty$
    { pop$ "" }
    { "[" swap$ * "]" * }
  if$
}
```

`paranthesify`

```
FUNCTION {paranthesify}
{
  duplicate$ empty$
    { pop$ "" }
    { "(" swap$ * ")" * }
  if$
}
```

`chop.word`

The function `chop.word` in the context '*sstr len str* `chop.word`' tries to remove given substring *sstr* of the length *len* from the beginning of the string *str*. It trims *str* only if first *len* symbols in *str* coinsides with *sstr*. See examples in Section 6.8.

```
⟨*sort | natbib⟩
INTEGERS { len }

FUNCTION {chop.word}
{ 's :=
  'len :=
  s #1 len substring$ =
    { s len #1 + global.max$ substring$ }
    's
  if$
}
⟨/sort | natbib⟩
```

`is.num` Currently not used.

⟨NB: Можно использовать для проверки года, извлечённого из `date`.⟩

```
⟨*debug⟩
FUNCTION {is.num}
{ chr.to.int$
  duplicate$ "0" chr.to.int$ < not
  swap$ "9" chr.to.int$ > not and
}

⟨/debug⟩
```

`extract.num` Currently not used.

```
⟨*debug⟩
FUNCTION {extract.num}
{ duplicate$ 't :=
  "" 's :=
  { t empty$ not }
  { t #1 #1 substring$
    t #2 global.max$ substring$ 't :=
    duplicate$ is.num
      { s swap$ * 's := }
      { pop$ "" 't := }
    if$
  }
  while$
  s empty$
    'skip$
    { pop$ s }
  if$
}

⟨/debug⟩
```

`tie.connect` Inserts unbreakable space between last two literals in the stack.

```
FUNCTION {tie.connect}
 {"~" swap$ * *
 }

```

`e.or.space.connect` Inserts space or unbreakable space between last two literals in the stack depending on the length of last literal.

```
FUNCTION {tie.or.space.connect}
{ duplicate$ text.length$ #3 <
    { "~" }
    { " " }
  if$
  swap$ * *
}

```

`n.dashify`

```
FUNCTION {n.dashify}
{ 't :=
  ""
    { t empty$ not }
    { t #1 #1 substring$ "-" =
        { t #1 #2 substring$ "--" = not
            { "--" *
              t #2 global.max$ substring$ 't :=
            }
            {   { t #1 #1 substring$ "-" = }
                { "-" *
                  t #2 global.max$ substring$ 't :=
```

```
              }
            while$
          }
        if$
      }
      { t #1 #1 substring$ *
        t #2 global.max$ substring$ 't :=
      }
    if$
  }
 while$
}

```

`multi.page.check` Returns 1 if the last literal (usually `page`) contains either '`-`', '`,`' or '`+`'; otherwise returns 0. Used in connection with `n.dashify`.

⟨NB: Заметим несогласованность функций `multi.page.check` и `n.dashify`. Последняя функция проверяет дефис, а первая ещё и минус и запятую.⟩

```
INTEGERS { multiresult }

FUNCTION {multi.page.check}
{ 't :=
  #0 'multiresult :=
    { multiresult not
      t empty$ not
      and
    }
    { t #1 #1 substring$
      duplicate$ "-" =
      swap$ duplicate$ "," =
      swap$ "+" =
      or or
        { #1 'multiresult := }
        { t #2 global.max$ substring$ 't := }
      if$
    }
  while$
  multiresult
}

```

6.5 Language-sensitive abbreviations

Declare language-sensitive abbreviations. We provide two versions of any abbreviation for unicode and non-Unicode styles. The language-sensitive functions push to the stack a string that depends of the the current value of the string `curlanguage`. It is set for every entry by `output.bibitem` function. Abbreviations for `russian`, `ukrainian` and `english` values of the string `curlanguage` are always provided, and sometimes for `french` and `german`.

`bbl.edby`

```
FUNCTION {bbl.edby}
{ curlanguage "english" =
   {"ed.\ by"}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"{\cyr\cyrp\cyrii\cyrd\ \cyrr\cyre\cyrd.}"}
⟨utf8⟩       {"під ред."}
      { curlanguage "russian" =
⟨!utf8⟩          {"{\cyr\cyrp\cyro\cyrd\ \cyrr\cyre\cyrd.}"}
⟨utf8⟩          {"под ред."}
         { curlanguage "german" =
            { "ed." }
            {"language is not defined: " curlanguage " in bbl.edby" * * warning$ "Ed.\ by"}
         if$}
      if$}
   if$}
if$}
```

bbl.compiler

```
FUNCTION {bbl.compiler}
{ curlanguage "english" =
   { "Compiler"}
   { curlanguage "german" =
      { "Hrsg." }
      { curlanguage "ukrainian" =
⟨!utf8⟩          {"{\cyr\CYRU\cyrk\cyrl.}"}
⟨utf8⟩          {"{Укл.}"}
         { curlanguage "russian" =
⟨!utf8⟩             {"{\cyr\CYRS\cyro\cyrs\cyrt.}"}
⟨utf8⟩             {"{Сост.}"}
            {"language is not defined: " curlanguage  " in bbl.compiler" * * warning$ "Compiler"}
         if$}
      if$}
   if$}
if$}
```

bbl.edition

```
FUNCTION {bbl.edition}
{ curlanguage "english" =
   {"ed."}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"{\cyr\cyrv\cyri\cyrd.}"}
⟨utf8⟩       {"{вид.}"}
      { curlanguage "russian" =
⟨!utf8⟩          {"{\cyr\cyri\cyrz\cyrd.}"}
⟨utf8⟩          {"{изд.}"}
         { curlanguage "german" =
            {" {aus.}" } %%% { "Aufl." } ??
            { curlanguage "italian" =
               {"edizione"}
```

```
                { curlanguage "french" =
                   {"\'{e}dition"}
                   {"language is not defined: " curlanguage  " in bbl.edition" * * warning$ "ed."}
                if$}
             if$}
          if$}
       if$}
    if$}
if$}

bbl.vvolume

FUNCTION {bbl.vvolume}
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
    {"Volume"}
    { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "\CYRT\cyro\cyrm" }
⟨utf8⟩       { "Том" }
       { curlanguage "german" =
          {"{Band}"} %%%% { "Volumen" }
          {"language is not defined: " curlanguage  " in bbl.vvolume" * * warning$ "Volume"}
       if$}
    if$}
if$}

  bbl.vvol

FUNCTION {bbl.vvol}
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
    {"Vol."}
    { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"\CYRT."}
⟨utf8⟩       {"T."}
       { curlanguage "german" =
          {"{Bd.}"} %%%% { "Vol." }
          {"language is not defined: " curlanguage  " in bbl.vvol" * * warning$ "Vol."}
       if$}
    if$}
if$}

bbl.iissue

FUNCTION {bbl.iissue}
{ curlanguage "english" =
    {"Issue"}
    { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRV\cyri\cyrp\cyru\cyrs\cyrk"}
⟨utf8⟩       {"Випуск"}
       { curlanguage "russian" =
⟨!utf8⟩          {"\CYRV\cyrery\cyrp\cyru\cyrs\cyrk"}
⟨utf8⟩          {"Выпуск"}
          { curlanguage "german" =
```

```
            {"{Heft}"} %%%% { "Ausgabe" }
            {"language is not defined: " curlanguage " in bbl.iissue" * * warning$ "Issue"}
         if$}
      if$}
   if$}
if$}

bbl.iiss

FUNCTION {bbl.iiss}
{ curlanguage "english" =
   {"Iss."}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRV\cyri\cyrp."}
⟨utf8⟩       {"Вип."}
      { curlanguage "russian" =
⟨!utf8⟩          {"\CYRV\cyrery\cyrp."}
⟨utf8⟩          {"Вып."}
         { curlanguage "german" =
            {"{H.}"}
            {"language is not defined: " curlanguage " in bbl.iiss" * * warning$ "Iss."}
         if$}
      if$}
   if$}
if$}

  bbl.of

FUNCTION {bbl.of}
{ curlanguage "english" =
   {"of"}
   { curlanguage "german" =
      { "von" }
      { curlanguage "ukrainian" =
⟨!utf8⟩          { "{\cyr\cyrii\cyrz}" }
⟨utf8⟩          { "{із}" }
         { curlanguage "russian" =
⟨!utf8⟩             { "{\cyr\cyri\cyrz}" }
⟨utf8⟩             { "{из}" }
            {"language is not defined: " curlanguage " in bbl.of" * * warning$ "of"}
         if$}
      if$}
   if$}
if$}

bbl.etal

FUNCTION {bbl.etal}
{ curlanguage "english" =
   {"et~al."}
   { curlanguage "german" =
      { "u.~a." }
```

```
      { curlanguage "ukrainian" =
⟨!utf8⟩          {"{\cyr\cyrt\cyra~\cyrii\cyrn.}"}
⟨utf8⟩          {"{та~ін.}"}
         { curlanguage "russian" =
⟨!utf8⟩             {"{\cyr\cyri~\cyrd\cyrr.}"}
⟨utf8⟩             {"и~др."}
            {"language is not defined: " curlanguage  " in bbl.etal" * * warning$ "et~al."}
         if$}
      if$}
   if$}
if$}

```

bbl.and

```
FUNCTION {bbl.and}
{ curlanguage "english" =
  {"and"}
   { curlanguage "german" =
      { "und" }
      { curlanguage "ukrainian" =
⟨!utf8⟩          {"{\cyrii}"}
⟨utf8⟩          {"i"}
         { curlanguage "russian" =
⟨!utf8⟩             {"{\cyri}"}
⟨utf8⟩             {"и"}
            { curlanguage "french" =
                {"et"}
                {"language is not defined: " curlanguage  " in bbl.and" * * warning$ "and"}
            if$}
         if$}
      if$}
   if$}
if$}

```

bbl.nnumber

```
FUNCTION {bbl.nnumber}
{ curlanguage "english" =
   {"Number"}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "\CYRN\cyro\cyrm\cyre\cyrr" }
⟨utf8⟩       { "{Номер}" }
      { curlanguage "german" =
         {"{Heft}"} %%% { "Anzahl" }
         {"language is not defined: " curlanguage  " in bbl.nnumber" * * warning$ "Number"}
      if$}
   if$}
if$}

```

bbl.number

```
FUNCTION {bbl.number}
```

```
{ curlanguage "english" =
   {"number"}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"{\cyr\cyrn\cyro\cyrm\cyre\cyrr}"}
⟨utf8⟩       {"{номер}"}
      { curlanguage "german" =
         {"{Heft}"} %%% { "anzahl" }???
         {"language is not defined: " curlanguage  " in bbl.number" * * warning$ "number"}
      if$}
   if$}
if$}

```

bbl.nr

```
FUNCTION {bbl.nr}
{ curlanguage "english" =
   {"no."}
   { curlanguage "italian" =
      { "no." }
      { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩          { "{\cyr\textnumero}" }
⟨utf8⟩          { "{№}" }
         { curlanguage "german" =
            {"{nu.}"} %%% { "an." }
            { curlanguage "french" =
                { "no." }
                {"language is not defined: " curlanguage " in bbl.nr" * * warning$ "no."}
            if$}
         if$}
      if$}
   if$}
if$}

```

bbl.nnr

```
FUNCTION {bbl.nnr}
{ curlanguage "english" =
   {"No."}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "{\cyr\textnumero}" }
⟨utf8⟩       { "{№}" }
      { curlanguage "german" =
         {"{H.}"} %%% { "an." }
         {"language is not defined: " curlanguage  " in bbl.nnr" * * warning$ "No."}
      if$}
   if$}
if$}

```

bbl.in

```
FUNCTION {bbl.in}
{ curlanguage "english" = curlanguage "german" = or
```

```
    {"in"}
    { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "{\cyr\cyrv}" }
⟨utf8⟩       { "{в}" }
       {"language is not defined: " curlanguage  " in bbl.in" * * warning$ "in"}
    if$}
if$}

```

bbl.iin Currently not used.

```
FUNCTION {bbl.iin}
{ curlanguage "english" = curlanguage "german" = or
    {"In"}
    { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "\CYRV" }
⟨utf8⟩       { "{B}" }
       {"language is not defined: " curlanguage  " in bbl.iin" * * warning$ "In"}
    if$}
if$}

```

bbl.pages

```
FUNCTION {bbl.pages}
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
    {"p."} %%% {"pp."}
    { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"{\cyr\cyrs.}"}
⟨utf8⟩       {"{c.}"}
       { curlanguage "german" =
          {"S."} %%%% { "s." }
          {"language is not defined: " curlanguage  " in bbl.pages" * * warning$ "p."}
       if$}
    if$}
if$}

```

bbl.page

```
FUNCTION {bbl.page}
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
    {"p."}
    { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"{\cyr\cyrs.}"}
⟨utf8⟩       {"{c.}"}
       { curlanguage "german" =
          {"S."} %%%% { "s." }
          {"language is not defined: " curlanguage  " in bbl.page" * * warning$ "p."}
       if$}
    if$}
if$}

```

bbl.ppages

```
FUNCTION {bbl.ppages}
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
   {"P."} %%%% { "Pp." }
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"{\cyr\CYRS.}"}
⟨utf8⟩       {"{С.}"}
      { curlanguage "german" =
         {"S."}
         {"language is not defined: " curlanguage " in bbl.ppages" * * warning$ "P."}
      if$}
   if$}
if$}
```

bbl.ppage

```
FUNCTION {bbl.ppage}
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
   {"P."}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"{\cyr\CYRS.}"}
⟨utf8⟩       {"{С.}"}
      { curlanguage "german" =
         {"S."}
         {"language is not defined: " curlanguage  " in bbl.ppage" * * warning$ "P."}
      if$}
   if$}
if$}
```

bbl.url Added in version 2016.07.07.

```
FUNCTION {bbl.url}
{ curlanguage "english" =
   {"Access mode"}
   { curlanguage "ukrainian" =
⟨!utf8⟩       { "{\CYRR\cyre\cyrzh\cyri\cyrm\ \cyrd\cyro\cyrs\cyrt\cyru\cyrp\cyru}" }
⟨utf8⟩       { "{Режим доступу}" }
      { curlanguage "russian" =
⟨!utf8⟩          { "{\CYRR\cyre\cyrzh\cyri\cyrm\ \cyrd\cyro\cyrs\cyrt\cyru\cyrp\cyra}" }
⟨utf8⟩          { "{Режим доступа}" }
         { curlanguage "german" =
            { "{online; abgerufen}" }
            { curlanguage "french" =
                { "Mode d'acc\`{e}s" }
                { "language is not defined: " curlanguage " in bbl.url" * * warning$ "online; accessed" }
            if$}
         if$}
      if$}
   if$}
if$}
```

bbl.urldate Added in version 2012.01.15.

```
FUNCTION {bbl.urldate}
```

```
{ curlanguage "english" =
   {"online; accessed"}
   { curlanguage "ukrainian" =
⟨!utf8⟩       { "{\cyrd\cyra\cyrt\cyra\ \cyrz\cyrv\cyre\cyrr\cyrn\cyre\cyrn\cyrn\cyrya}" }
⟨utf8⟩       { "{дата звернення}" }
      { curlanguage "russian" =
⟨!utf8⟩          { "{\cyrd\cyra\cyrt\cyra\ \cyro\cyrb\cyrr\cyra\cyrshch\cyre\cyrn\cyri\cyrya}" }
⟨utf8⟩          { "{дата обращения}" }
         { curlanguage "german" =
            { "{online; abgerufen}" }
            { curlanguage "french" =
               { "{en ligne; acc\'{e}d\'{e}}" }
               { "language is not defined: " curlanguage " in bbl.urldate" * * warning$ "online; accessed" }
            if$}
         if$}
      if$}
   if$}
if$}
```

bbl.techreport

```
FUNCTION {bbl.techreport}
{ curlanguage "english" =
   { "Rep." }
   { curlanguage "german" =
      { "Bericht" }
      { curlanguage "russian" =
⟨!utf8⟩          { "{\cyr\CYRO\cyrt\cyrch\cyre\cyrt}" }
⟨utf8⟩          { "{Отчет}" }
         { "language is not defined: " curlanguage " in bbl.techrep" * * warning$ "Rep." }
      if$}
   if$}
if$}
```

bbl.mathesis

```
FUNCTION {bbl.mathesis}
{ curlanguage "english" =
   { "Master's thesis" }
   { curlanguage "german" =
      { "diss.~mag." }
      { curlanguage "russian" =
⟨!utf8⟩        { "{\cyr\cyrk\cyrv\cyra\cyrl\cyri\cyrf\cyri\cyrk\cyra\cyrc\cyri"
⟨!utf8⟩          "\cyro\cyrn\cyrn\cyra\cyrya\ \cyrr\cyra\cyrb\cyro\cyrt\cyra\ " *
⟨!utf8⟩          "\cyrm\cyra\cyrg\cyri\cyrs\cyrt\cyrr\cyra}" * }
⟨utf8⟩          { "{квалификационная работа магистра}" }
         { "language is not defined: " curlanguage " in bbl.mthesis" * * warning$ "Master's thesis" }
      if$}
   if$}
if$}
```

bbl.phdthesis

```
FUNCTION {bbl.phdthesis}
{ curlanguage "english" =
   { "Ph.\,D. thesis" }
   { curlanguage "german" =
      { "diss.~Ph.\,D." }
      { curlanguage "russian" =
⟨!utf8⟩          { "{\cyr\cyrd\cyri\cyrs.\ \ldots\ \cyrk\cyra\cyrn\cyrd. "
⟨!utf8⟩            "\cyrn\cyra\cyru\cyrk}" * }
⟨utf8⟩          { "{дис.\ \ldots\ канд.\ наук}" }
         { curlanguage "french" =
           { "th\‘{e}se de doctorat" }
           { "language is not defined: " curlanguage " in bbl.phdthesis" * * warning$ "Ph.\,D. thesis" }
         if$}
      if$}
   if$}
if$}
```

bbl.docthesis

```
FUNCTION {bbl.docthesis}
{ curlanguage "english" =
   { "dr.\,sci. dissertation" }
   { curlanguage "german" =
      { "diss.~dr." }
      { curlanguage "russian" =
⟨!utf8⟩          { "{\cyr\cyrd\cyri\cyrs.\ \ldots\ \cyrd-\cyrr\cyra\ "
⟨!utf8⟩            "\cyrn\cyra\cyru\cyrk}" * }
⟨utf8⟩          { "{дис.\ \ldots\ д-ра наук}" }
         { "language is not defined: " curlanguage " in bbl.docthesis" * * warning$ "Dr.\,Sci. dissertation"
      if$}
   if$}
if$}
```

bbl.nnoaddress

```
FUNCTION {bbl.nnoaddress}
{ curlanguage "english" =
   { "S.\ l." }
   { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYRB.\ \cyrm.}" }
⟨utf8⟩       { "{Б.\ м.}" }
      { "language is not defined: " curlanguage " in bbl.nnoaddress" * * warning$ "S.\ l." }
   if$}
if$}
```

bbl.nopublisher

```
FUNCTION {bbl.nopublisher}
{ curlanguage "english" =
   { "s.\ n." }
```

```
    { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\cyrb.\ \cyri.}" }
⟨utf8⟩       { "{б.\ и.}" }
        { "language is not defined: " curlanguage " in bbl.nnopublisher" * * warning$ "s.\ n." }
    if$}
if$}

bbl.nnopublisher

FUNCTION {bbl.nnopublisher}
{ curlanguage "english" =
    { "S.\ n." }
    { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYRB.\ \cyri.}" }
⟨utf8⟩       { "{Б.\ и.}" }
        { "language is not defined: " curlanguage " in bbl.nnopublisher" * * warning$ "S.\ n." }
    if$}
if$}

  bbl.media.text

FUNCTION {bbl.media.text}
{ curlanguage "english" =
    { "Text" }
    { curlanguage "russian" = curlanguage "ukrainian" = or
⟨!utf8⟩       { "{\cyr\CYRT\cyre\cyrk\cyrs\cyrt}" }
⟨utf8⟩       { "{Текст}" }
        { "language is not defined: " curlanguage " in bbl.media" * * warning$ "Text" }
    if$}
if$}

l.media.eresource

FUNCTION {bbl.media.eresource}
{ curlanguage "english" =
    { "Electronic resource" }
    { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYREREV\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyrery\cyrishrt\ "
⟨!utf8⟩         "\cyrr\cyre\cyrs\cyru\cyrr\cyrs}" * }
⟨utf8⟩       { "{Электронный ресурс}" }
        { curlanguage "ukrainian" =
⟨!utf8⟩         { "{\cyr\CYRE\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyri\cyrishrt\ "
⟨!utf8⟩           "\cyrr\cyre\cyrs\cyru\cyrr\cyrs}" * }
⟨utf8⟩         { "{Електронний ресурс}" }
          { "language is not defined: " curlanguage " in bbl.media" * * warning$ "Electronic resource" }
        if$}
    if$}
if$}

bbl.media.online

FUNCTION {bbl.media.online}
```

```
{ curlanguage "english" =
   { "Electronic resource online" }
   { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYREREV\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyrery\cyrishrt\ "
⟨!utf8⟩         "\cyrr\cyre\cyrs\cyru\cyrr\cyrs\ \cyro\cyrn\cyrl\cyra\cyrishrt\cyrn}" * }
⟨utf8⟩       { "{Электронный ресурс онлайн}" }
      { curlanguage "ukrainian" =
⟨!utf8⟩         { "{\cyr\CYRE\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyri\cyrishrt\ "
⟨!utf8⟩           "\cyrr\cyre\cyrs\cyru\cyrr\cyrs\ \cyro\cyrn\cyrl\cyra\cyrishrt\cyr}" * }
⟨utf8⟩         { "{Електронний ресурс онлайн}" }
        { "language is not defined: " curlanguage " in bbl.media" * * warning$ "Electronic resource" }
      if$}
   if$}
if$}
```

bbl.chief

```
FUNCTION {bbl.chief}
{ curlanguage "english" =
   { "chief" }
   { curlanguage "russian" =
⟨!utf8⟩       { "\cyrr\cyru\cyrk." }
⟨utf8⟩       { "{рук.}" }
      { curlanguage "ukrainian" =
⟨!utf8⟩         { "\cyrr\cyru\cyrk." }
⟨utf8⟩         { "{рук.}" }
        { "language is not defined: " curlanguage " in bbl.chief" * * warning$ "chief" }
      if$}
   if$}
if$}
```

bbl.executor

```
FUNCTION {bbl.executor}
{ curlanguage "english" =
   { "executor" }
   { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\cyri\cyrs\cyrp\cyro\cyrl\cyrn.}" }
⟨utf8⟩       { "{исполн.}" }
      { curlanguage "ukrainian" =
⟨!utf8⟩         { "{\cyr\cyrv\cyri\cyrk\cyro\cyrn\cyra\cyrv\cyre\cyrc\cyrsftsn}" }
⟨utf8⟩         { "{виконавець}" }
        { "language is not defined: " curlanguage " in bbl.executor" * * warning$ "executor" }
      if$}
   if$}
if$}
```

bbl.media

```
FUNCTION {bbl.media}
{ media "online" =
   { bbl.media.online }
```

```
    { media "eresource" =
      { bbl.media.eresource }
      { bbl.media.text }
    if$}
if$}

bbl.req
FUNCTION {bbl.req}
{
  curlanguage "english" =
    { "req." }
    { curlanguage "german" =
      { "ang." }
      { curlanguage "russian" =
⟨!utf8⟩         { "{\cyr\cyrz\cyra\cyrya\cyrv\cyrl.}" }
⟨utf8⟩         { "{заявл.}" }
        { "language is not defined: " curlanguage " in bbl.req" * * warning$ "req" }
      if$
      }
    if$
    }
  if$
}

bbl.publ
FUNCTION {bbl.publ}
{
  curlanguage "english" =
    { "publ." }
    { curlanguage "german" =
      { "ausg." }
      { curlanguage "russian" =
⟨!utf8⟩         { "{\cyr\cyro\cyrp\cyru\cyrb\cyrl.}" }
⟨utf8⟩         { "{опубл.}" }
        { "language is not defined: " curlanguage " in bbl.publication" * * warning$ "publication" }
      if$
      }
    if$
    }
  if$
}

bbl.priority
FUNCTION {bbl.priority}
{
  curlanguage "english" =
    { "priority" }
    { curlanguage "german" =
      { "Prioritat" }
```

```
      { curlanguage "russian" =
⟨!utf8⟩        { "{\cyr\cyrp\cyrr\cyri\cyro\cyrr\cyri\cyrt\cyre\cyrt}" }
⟨utf8⟩        { "{приоритет}" }
        { "language is not defined: " curlanguage " in bbl.priority" * * warning$ "priority" }
      if$
      }
    if$
    }
  if$
}

```

bbl.jan
bbl.feb
bbl.mar
bbl.apr
bbl.may
bbl.jun
bbl.jul
bbl.aug
bbl.sep
bbl.oct
bbl.nov
bbl.dec

New in version 1.2k.

```
FUNCTION {bbl.jan}
{ curlanguage "english" =
   {"Jan."}
   { curlanguage "ukrainian" =
⟨!utf8⟩      {"\CYRS\cyrii\cyrch."}
⟨utf8⟩      {"Січ."} % Січень
      { curlanguage "russian" =
⟨!utf8⟩         { "\CYRYA\cyrn\cyrv." }
⟨utf8⟩         { "Янв." }
         { curlanguage "german" =
            { "Jan." } % Januar
            { "language is not defined: bbl.jan for " curlanguage * warning$ "Jan." }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.feb}
{ curlanguage "english" =
   {"Feb."}
   { curlanguage "ukrainian" =
⟨!utf8⟩      {"\CYRL\cyryu\cyrt."}
⟨utf8⟩      {"Лют."} % Лютий
      { curlanguage "russian" =
⟨!utf8⟩         { "\CYRF\cyre\cyrv\cyrr." }
⟨utf8⟩         { "Фев." }
         { curlanguage "german" =
            {"Feb."} % Februar
            {"language is not defined: bbl.feb for " curlanguage * warning$ "Feb."}
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.mar}
{ curlanguage "english" =
   {"Mar."}
   { curlanguage "ukrainian" =
```

```
⟨!utf8⟩       {"\CYRB\cyre\cyrr."}
⟨utf8⟩       {"Бер."} % Березень
       { curlanguage "russian" =
⟨!utf8⟩             { "\CYRM\cyra\cyrr\cyrt" }
⟨utf8⟩             { "Март" }
          { curlanguage "german" =
             {"März"} % März
             {"language is not defined: bbl.mar for " curlanguage * warning$ "Mar."}
          if$}
       if$}
    if$}
if$}

FUNCTION {bbl.apr}
{ curlanguage "english" =
    {"Apr."}
    { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRK\cyrv\cyrii\cyrt."}
⟨utf8⟩       {"Квіт."} % квітень
       { curlanguage "russian" =
⟨!utf8⟩             { "\CYRA\cyrp\cyrr." }
⟨utf8⟩             { "Апр." }
          { curlanguage "german" =
             {"Apr."} % April
             { "language is not defined: bbl.apr for " curlanguage * warning$ "Apr." }
          if$}
       if$}
    if$}
if$}

FUNCTION {bbl.may}
{ curlanguage "english" =
    {"May"}
    { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRT\cyrr\cyra\cyrv."}
⟨utf8⟩       {"Трав."} % травень
       { curlanguage "russian" =
⟨!utf8⟩             { "\CYRM\cyra\cyrishrt" }
⟨utf8⟩             { "Май" }
          { curlanguage "german" =
             {"Mai"}
             { "language is not defined: bbl.may for " curlanguage * warning$ "May" }
          if$}
       if$}
    if$}
if$}

FUNCTION {bbl.jun}
{ curlanguage "english" =
    {"June"}
```

```
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRCH\cyre\cyrr."}
⟨utf8⟩       {"Чер."} % червень
      { curlanguage "russian" =
⟨!utf8⟩             { "\CYRI\cyryu\cyrn\cyrsftsn" }
⟨utf8⟩             { "Июнь" }
         { curlanguage "german" =
            {"Juni"}
            { "language is not defined: bbl.jun for " curlanguage * warning$ "June" }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.jul}
{ curlanguage "english" =
   {"July"}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRL\cyri\cyrp."}
⟨utf8⟩       {"Липень"} %Липень
      { curlanguage "russian" =
⟨!utf8⟩             { "\CYRI\cyryu\cyrl\cyrsftsn" }
⟨utf8⟩             { "Июль" }
         { curlanguage "german" =
            {"Juli"}
            { "language is not defined: bbl.jul for " curlanguage * warning$ "July" }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.aug}
{ curlanguage "english" =
   {"Aug."}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRS\cyre\cyrr."}
⟨utf8⟩       {"Серпень"} % Серпень
      { curlanguage "russian" =
⟨!utf8⟩             { "\CYRA\cyrv\cyrg\." }
⟨utf8⟩             { "Авг." }
         { curlanguage "german" =
            {"Aug."} % August
            { "language is not defined: bbl.aug for " curlanguage * warning$ "Aug." }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.sep}
{ curlanguage "english" =
```

```
   {"Sep."}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRV\cyre\cyrr."}
⟨utf8⟩       {"Вер."} % вересень
      { curlanguage "russian" =
⟨!utf8⟩             { "\CYRS\cyre\cyrn\cyrt." }
⟨utf8⟩             { "Сент." }
         { curlanguage "german" =
            {"Sep."} % September
            { "language is not defined: bbl.sep for " curlanguage * warning$ "Sep." }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.oct}
{ curlanguage "english" =
   {"Oct."}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRZH\cyro\cyrn."}
⟨utf8⟩       {"Жов."} % жовтень
      { curlanguage "russian" =
⟨!utf8⟩             { "\CYRO\cyrk\cyrt." }
⟨utf8⟩             { "Окт." }
         { curlanguage "german" =
            {"Okt."} % Oktober
            { "language is not defined: bbl.oct for " curlanguage * warning$ "Oct." }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.nov}
{ curlanguage "english" =
   {"Nov."}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRL\cyri\cyrs."}
⟨utf8⟩       {"Лис."} % листопад
      { curlanguage "russian" =
⟨!utf8⟩             { "\CYRN\cyro\cyrya\cyrb." }
⟨utf8⟩             { "Нояб." }
         { curlanguage "german" =
            {"Nov."} % November
            { "language is not defined: bbl.nov for " curlanguage * warning$ "Nov." }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.dec}
```

```
{ curlanguage "english" =
   {"Dec."}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"\CYRG\cyrr\cyru\cyrd."}
⟨utf8⟩       {"Груд."} % грудень
      { curlanguage "russian" =
⟨!utf8⟩             { "\CYRD\cyre\cyrk." }
⟨utf8⟩             { "Дек." }
         { curlanguage "german" =
            {"Dez."} % Dezember
            { "language is not defined: bbl.dec for " curlanguage * warning$ "Dec." }
         if$}
      if$}
   if$}
if$}
```

bbl.arxiv
bbl.jstor
bbl.pubmed
bbl.googlebooks
bbl.hdl

New in version 1.2k.

```
FUNCTION {bbl.arxiv}
{ curlanguage "english" =
   { "ArXiv" }
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"ArXiv"}
⟨utf8⟩       {"ArXiv"}
      { curlanguage "russian" =
⟨!utf8⟩          { "ArXiv" }
⟨utf8⟩          { "ArXiv" }
         { curlanguage "german" =
            { "ArXiv" }
            { "language is not defined: bbl.arxiv for " curlanguage * warning$ "ArXiv" }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.jstor}
{ curlanguage "english" =
   { "JSTOR" }
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"JSTOR"}
⟨utf8⟩       {"JSTOR"}
      { curlanguage "russian" =
⟨!utf8⟩          { "JSTOR" }
⟨utf8⟩          { "JSTOR" }
         { curlanguage "german" =
            { "JSTOR" }
            { "language is not defined: bbl.jstor for " curlanguage * warning$ "JSTOR" }
         if$}
      if$}
   if$}
if$}

```

```
FUNCTION {bbl.pubmed}
{ curlanguage "english" =
   { "PubMed" }
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"PubMed"}
⟨utf8⟩       {"PubMed"}
      { curlanguage "russian" =
⟨!utf8⟩          { "PubMed" }
⟨utf8⟩          { "PubMed" }
         { curlanguage "german" =
            { "PubMed" }
            { "language is not defined: bbl.pubmed for " curlanguage * warning$ "PubMed" }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.googlebooks}
{ curlanguage "english" =
   { "Google Books" }
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"Google \CYRK\cyrn\cyri\cyrg\cyri"}
⟨utf8⟩       {"Google Книги"}
      { curlanguage "russian" =
⟨!utf8⟩          { "Google \CYRK\cyrn\cyri\cyrg\cyri" }
⟨utf8⟩          { "Google Книги" }
         { curlanguage "german" =
            { "Google Books" }
            { "language is not defined: bbl.googlebooks for " curlanguage * warning$ "Google Books" }
         if$}
      if$}
   if$}
if$}

FUNCTION {bbl.hdl}
{ curlanguage "english" =
   { "Handle.Net" }
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"Handle.Net"}
⟨utf8⟩       {"Handle.Net"}
      { curlanguage "russian" =
⟨!utf8⟩          { "Handle.Net" }
⟨utf8⟩          { "Handle.Net" }
         { curlanguage "german" =
            { "Handle.Net" }
            { "language is not defined: bbl.hdl for " curlanguage * warning$ "Handle.Net" }
         if$}
      if$}
   if$}
if$}
```

6.6 Aliases

Aliases to some fields are introduced with the help of *field1* `.or.` *field2* functions.

`dress.or.location` Pushes `address` field if not empty; otherwise pushes `location` value even if it is empty.

```
%FUNCTION {address.or.location}
%{
%  address empty$
%    { location empty$
%      'skip$
%      { location }
%    if$ }
%    { address }
%  if$
%}
FUNCTION {address.or.location}
{
  address empty$
    { location }
    { address }
  if$
}
```

`itycode.or.number` Pushed `specialitycode` value if not empty; otherwise returns `number` value even if the latter is empty.

```
FUNCTION {specialitycode.or.number}
{
  specialitycode empty$
    { number }
    { specialitycode }
  if$
}
```

`itution.or.school` Pushes `institution` value if not empty; otherwise returns `school` value even if the latter is empty.

```
FUNCTION {institution.or.school}
{
  institution empty$
    { school}
    { institution }
  if$
}
```

6.7 Formatting dates

⟨NB: NEEDS to be located before natbib labels. This is experimental section. Needs to be upgraded.⟩

`date.to.year` New in version 1.2k. Extracts year from date. Currently, only the date of the form either `YYYY-MM-DD`, or `YYYY-MM`, or `YYYY` can be processed as expected. Returns `date` if it is empty or has wrong format.

```
FUNCTION {date.to.year}
{
  date empty$
    { date }
    {
⟨*debug⟩
      "date.to.year::: date in " cite$ * " =" * date * warning$
      "date.to.year::: length of date is " date text.length$ int.to.str$ * warning$
⟨/debug⟩
      date text.length$ #3 >
        {
          date #1 #4 substring$ 'y :=
⟨*debug⟩
          "::::::: y=" y * warning$
⟨/debug⟩
          y
        }
        {
          "wrong format of date in " cite$ * ": date=" * date * warning$
          date
        }
      if$
    }
  if$
}

```

`date.to.month` New in version 1.2k. Extracts month from date. Returns `date` if it is empty or has wrong format.

```
FUNCTION {date.to.month}
{
  date empty$
    { date }
    {
⟨*debug⟩
      "date.to.month::: date in " cite$ * " =" * date * warning$
      "date.to.month::: length of date is " date text.length$ int.to.str$ * warning$
⟨/debug⟩
      date text.length$ #6 >
        {
          date #6 #2 substring$ 'm :=
⟨*debug⟩
          "::::::: m=" m * warning$
⟨/debug⟩
          m
        }
        {
          "wrong format of date in " cite$ * ": date=" * date * warning$
          date
        }
      if$
```

```
    }
  if$
}
```

`date.to.day` New in version 1.2k. Extracts year from date. Returns `date` if it is empty or has wrong format.

```
FUNCTION {date.to.day}
{
  date empty$
    { date }
    {
⟨*debug⟩
      "date.to.day::: date in " cite$ * " =" * date * warning$
      "date.to.day::: length of date is " date text.length$ int.to.str$ * warning$
⟨/debug⟩
      date text.length$ #10 =
        {
          date #9 #2 substring$ 'd :=
⟨*debug⟩
          "::::::: d=" d * warning$
⟨/debug⟩
          d
        }
        {
          "wrong format of date in " cite$ * ": date=" * date * warning$
          date
        }
      if$
    }
  if$
}
```

`r.or.date.to.year` New in version 1.2k. Returns `year` if not empty; otherwise call `date.to.year`.
⟨№: ВОЗМОЖНО, ПОДОБНЫЕ ФУНКЦИИ ЛУЧШЕ назвать `this.year`.⟩

```
FUNCTION {year.or.date.to.year}
{
  year empty$
    {
⟨*debug⟩
      "year.or.date.to.year::: empty year in " cite$ * warning$
⟨/debug⟩
      date.to.year
    }
    { year }
  if$
}
```

`format.month` Reads `month` field and translate standard English abbreviation of months (as defined by

`jan`, `feb`, etc. macros below) to current language. ⟨№: НАДО ИЗОБРЕСТИ ЧТО-ТО ВРОДЕ `month.or.date.to.month` (`this.month`).⟩

```
FUNCTION {format.month}
{ month empty$
    { "" }
    { month "Jan." =
        { bbl.jan }
    { month "Feb." =
        { bbl.feb }
    { month "Mar." =
        { bbl.mar }
    { month "Apr." =
        { bbl.apr }
    { month "May" =
        { bbl.may }
    { month "Jun." =
        { bbl.jun }
    { month "Jul." =
        { bbl.jul }
    { month "Aug." =
        { bbl.aug }
    { month "Sep." =
        { bbl.sep }
    { month "Oct." =
        { "Окт." }
    { month "Nov." =
        { bbl.nov }
    { month "Dec." =
        { bbl.dec }
        {
          "unknown month in " cite$ * warning$
          month
        }
        if$}if$}if$}if$}if$}if$}if$}if$}if$}if$}if$}if$}
    if$
}
```

`format.date`

```
FUNCTION {format.date}
%{ year empty$
{ year.or.date.to.year empty$
    { month empty$
        { "" }
        { "there's a month but no year in " cite$ * warning$
          format.month
        }
      if$
    }
    { month empty$
%       'year
```

```
%        { year ". \BibDash " format.month * * }
        'year.or.date.to.year
        { year.or.date.to.year ". \BibDash " format.month * * }
      if$
    }
  if$
}
```

6.8 Formatting names

Declare functions to format separate elements of a bibliographic reference.

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 (which trims first name to first letter) when option `utf8` is in effect; in the latter caase `ff` primitive is called instead.

```
INTEGERS { nameptr namesleft numnames }
```

`fmt.names.first` `fmt.names.three` `fmt.names.all`

New version of functions formating names. Function name indicates number of persons printed. These functions look for last literal in the stack which should not be empty; hence, they should be called by other functions that checks if the last literal is empty.

```
FUNCTION {fmt.names.first}
{ #1
⟨strict&!utf8⟩   "{vv~}{ll}{~jj}{,~f.}"
⟨strict & utf8⟩   "{vv~}{ll}{~jj}{,~ff}"
⟨!strict&!utf8⟩   "{vv~}{ll}{~jj}{~f.}"
⟨!strict & utf8⟩   "{vv~}{ll}{~jj}{~ff}"
  format.name$
}

FUNCTION {fmt.names.three}
{
  's :=
  #1 'nameptr :=
  s num.names$ 'numnames :=
  numnames 'namesleft :=
  { namesleft #0 > }
  { s nameptr
⟨strict&!utf8⟩       "{vv~}{ll}{~jj}{,~f.}"
⟨strict & utf8⟩       "{vv~}{ll}{~jj}{,~ff}"
⟨!strict&!utf8⟩       "{vv~}{ll}{~jj}{~f.}"
⟨!strict & utf8⟩       "{vv~}{ll}{~jj}{~ff}"
    format.name$ 't :=
    nameptr #1 >
      { nameptr #4 = numnames #4 > and
          { "others" 't :=
            #1 'namesleft :=
          }
```

```
          'skip$
        if$
        namesleft #1 >
          { ", " * t * }
          { t "others" = t "~others" = or
⟨!strict⟩                { " " * bbl.etal * }
⟨strict⟩                { " " * bbl.etal bracketise *}
              { ", " * t * }
          if$
          }
        if$
      }
      't
    if$
    nameptr #1 + 'nameptr :=
    namesleft #1 - 'namesleft :=
  }
  while$
}

FUNCTION {fmt.names.all}
{ 's :=
  #1 'nameptr :=
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
    { s nameptr
      %"{vv~}{ll}" format.name$ 't :=
⟨strict&!utf8⟩       "{vv~}{ll}{~jj}{,~f.}"
⟨strict & utf8⟩       "{vv~}{ll}{~jj}{,~ff}"
⟨!strict&!utf8⟩       "{vv~}{ll}{~jj}{~f.}"
⟨!strict & utf8⟩       "{vv~}{ll}{~jj}{~ff}"
      format.name$ 't :=
      nameptr #1 >
        { namesleft #1 >
            { ", " * t * }
            { numnames #2 >  curlanguage "english" = and
                { "," * }
                'skip$
              if$
              t "others" = t "~others" = or
⟨!strict⟩                { " " * bbl.etal * }
⟨strict⟩                { " " * bbl.etal bracketise *}
                { " " bbl.and " " * * * t * }
              if$
            }
          if$
        }
        't
      if$
```

```
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}
```

6.9 Formatting names (cont.)

`format.author` Formats a list of authors for the heading part of a bibliographic record by appying either `fmt.names.first`, `fmt.names.three` or `fmt.names.all` to the field `author`, if it is not empty; otherwise it pushes empty `author`. This allows making a check as in the case of `format.author "author" output.check`.

⟨NB: Однако заметим, что все другие функции `format...` пустое поле замещают пустой строкой.⟩

```
%FUNCTION {format.author}
%{ author empty$
%    %%%{ "" }  % < v.1.2k
%    { author } % v.1.2k
%    %{ author format.names emphasize} % 1st if strict, <=3 otherwise
%%<long>    { author fmt.names.all emphasize}
%%<!long&strict>    { author fmt.names.first emphasize}
%%<!long&!strict>    { author fmt.names.three emphasize}
%  if$
%}
FUNCTION {format.author}
{
  author empty$
    { author } %%%'skip$
⟨*long⟩
    { author fmt.names.all }
⟨/long⟩
⟨*!long⟩
    {author num.names$ #4 <
      {
⟨strict⟩        author fmt.names.first
⟨!strict⟩        author fmt.names.three
      }
      { "" } %%%'skip$
    if$}
⟨/!long⟩
  if$
}
```

`ormat.bookauthors` Is used only once by `bookauthor.head` called in `inbook` entry. ⟨NB: Not used anymore!⟩

```
⟨*debug⟩
%FUNCTION {format.bookauthors}
%{ bookauthor empty$
%    { "" }
%    { bookauthor format.names}% cuts to 4 persons if |!strict| option
```

```
  %  if$
  %}
  %
⟨/debug⟩
```

ormat.author.rest Formats `author` to be placed after a slash in the zone of responsibiloty of a bibliographic record. In contrast to `output.author.rest` does not check number of items in the `author` field; it is recommended to use `output.author.rest` instead whenever possible. ⟨NB: Not used any more.⟩

```
⟨*debug⟩
FUNCTION {format.author.rest}
{
⟨*long⟩
  %%% Does this work?
  %%%skip$     % this seemed to work
  %%% ""       % this seemed to work
⟨/long⟩
⟨*!long⟩
  author empty$
    %%%{ "" } % < v.1.2k
    { author } % v.1.2k
    { author fmt.names.three }
  if$
⟨/!long⟩
}

⟨/debug⟩
```

.bookauthors.rest

```
FUNCTION {format.bookauthors.rest}
{ bookauthor empty$
    { "" }
⟨long⟩    { bookauthor fmt.names.all emphasize }
⟨!long⟩    { bookauthor fmt.names.three emphasize }
  if$
}

```

rmat.editors.rest

```
FUNCTION {format.editors.rest}
{ editor empty$
    { "" }
⟨long⟩    { bbl.edby "\ " * editor fmt.names.all * }
⟨!long⟩    { bbl.edby "\ " * editor fmt.names.three * }
  if$
}

```

format.chief.rest Formats `editor` field for `report` and `techreport` entries.

```
FUNCTION {format.chief.rest}
{ editor empty$
    { "" }
⟨long⟩    { bbl.chief "\ " * editor fmt.names.all * }
```

```
⟨!long⟩    { bbl.chief "\ " * editor fmt.names.three * }
  if$
}

mat.executor.rest

FUNCTION {format.executor.rest}
{ author empty$
    { "" }
⟨long⟩    { bbl.executor ": " * author fmt.names.all * }
⟨!long⟩    { bbl.executor ": " * author fmt.names.three * }
  if$
}

mat.compiler.rest

FUNCTION {format.compiler.rest}
{ compiler empty$
    { "" }
⟨long⟩    { bbl.compiler "\ " * compiler fmt.names.all * }
⟨!long⟩    { bbl.compiler "\ " * compiler fmt.names.three * }
  if$
}
```

6.10 Formatting natbib keys

⟨∗natbib⟩

fmt.names.brief

Formats all names like `fmt.names.all` except that first name of every person is dropped (dropped `ff` and `f.` modifiers).

⟨№: Вероятно, нужно сократить число персон до 3х максимум. Что говорит ГОСТ? Где используется это? В метке ссылки это расширенный список авторов после краткого списка и года в необязательном аргументе `\bibitem`.⟩

```
FUNCTION {fmt.names.brief}
{ 's :=
  #1 'nameptr :=
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
    { s nameptr "{vv~}{ll}" format.name$ 't :=
      nameptr #1 >
        {
          namesleft #1 >
            { ", " * t * }
            {
              numnames #2 >  curlanguage "english" = and
                { "," * }
                'skip$
              if$
              t "others" =  t "~others" =  or
```

```
1700                { " " bbl.etal * * }
1701                { " " bbl.and " " * * * t * }
1702              if$
1703            }
1704          if$
1705        }
1706        't
1707      if$
1708      nameptr #1 + 'nameptr :=
1709      namesleft #1 - 'namesleft :=
1710    }
1711  while$
1712 }
1713
```

`format.names.key` Declares function to go to left part of optional argument of `\bibitem` in the bibstyles generated with the option `natbib`. It cuts the list of person to 2 at most and drops first name of every person.

```
1714 FUNCTION {format.names.key}
1715 { 's :=
1716   set.language %%% уже вызвана в output.bibitem, но вроде бы нужна и здесь
1717   s #1 "{vv~}{ll}" format.name$
1718   s num.names$ duplicate$
1719   #2 >
1720     { pop$ " " bbl.etal * * }
1721     { #2 <
1722         'skip$
1723         { s #2 "{ff }{vv }{ll}{ jj}" format.name$ "others" =
1724             { " " bbl.etal * * }
1725             { " " bbl.and " " * * * s #2 "{vv~}{ll}" format.name$ * }
1726           if$
1727         }
1728       if$
1729     }
1730   if$
1731 }
1732
```

`format.key` Substitute an empty last literal in the stack (usually, `author`) with the `key` field if provided. Used as a heading of bibliographic record of `author` is empty.

```
1733 FUNCTION {format.key}
1734 { empty$
1735     { key field.or.null }
1736     { "" }
1737   if$
1738 }
1739
```

`author.key.label` Composes a key to be used as a reference label with natbib styles. If `author` field is empty an attempt is made to retrieve `key` field. If it is also empty 3 first letters are retreived from the citation key `cite$`. If `author` field is not empty `format.names.key` defined above is called.

```
1740 FUNCTION {author.key.label}
1741 { author empty$
```

```
    { key empty$
        { cite$ #1 #3 substring$ }
        'key
      if$
    }
    { author format.names.key }
  if$
}
```

.editor.key.label
y.organization.label
y.organization.label

These functions operate similarly but engage a different set of fields. They are called below by calc.short.list.

```
FUNCTION {author.editor.key.label}
{ author empty$
    { editor empty$
        { key empty$
            { cite$ #1 #3 substring$ }
            %'key          %% causes lost of year
            { "{}" key * } %% Bug in bibtex8 ??
          if$
        }
        { editor format.names.key }
      if$
    }
    { author format.names.key }
  if$
}

FUNCTION {author.key.organization.label}
{ author empty$
    { key empty$
        { organization empty$
            { cite$ #1 #3 substring$ }
            { "The " #4 organization chop.word #3 text.prefix$ }
          if$
        }
        'key
      if$
    }
    { author format.names.key }
  if$
}

FUNCTION {editor.key.organization.label}
{ editor empty$
    { key empty$
        { organization empty$
            { cite$ #1 #3 substring$ }
            { "The " #4 organization chop.word #3 text.prefix$ }
          if$
        }
        'key
```

```
      if$
    }
    { editor format.names.key }
  if$
}
```

`calc.short.list` Calculates `short.list` for `\bibitem` label in natbib styles depending on the type of entry.

⟨NB: Нужна внимательная ревизия логики вычислений. В общем-то, она имеет значения в особом случае, когда поле `author` пустое. ⟩

```
FUNCTION {calc.short.list}
{ type$ "book" = type$ "inbook" = or
    'author.editor.key.label
    { type$ "proceedings" =
        'editor.key.organization.label
        { type$ "manual" =
            'author.key.organization.label
            'author.key.label
          if$
        }
      if$
    }
  if$
  'short.list :=
}
```

`calc.label`

⟨NB: Вопреки названию, вычисляет только часть метки в [] для `\bibitem`. Скобка после года не закрыта, так как еще может быть добавлено a, b, ... Лучше перенести весь код `output.bibitem`. ⟩

```
FUNCTION {calc.label}
{ calc.short.list
  short.list
  "("
  *
%  year duplicate$ empty$
  year.or.date.to.year duplicate$ empty$
  short.list key field.or.null = or
     { pop$ "" }
     'skip$
  if$
  *
  'label :=
}
```

`calc.long.list` In case of `natbib` option, we need `calc.long.list` to compose `output.bibitem`, and the latter, in its turn, requires some more functions.

```
FUNCTION {calc.long.list} %% called 1 time only
{ type$ "book" = type$ "inbook" = or
    %'format.author.editor.brief
```

```
    { author empty$
        { editor empty$
            { "" }
            { editor fmt.names.brief }
          if$
        }
        { author fmt.names.brief }
      if$
    }
    { type$ "proceedings" =
        %'format.editor.brief
        { editor empty$
            { "" }
            { editor fmt.names.brief }
          if$
        }
        %'format.author.brief
        { author empty$
            { "" }
            { author fmt.names.brief }
          if$
        }
      if$
    }
  if$
}

```

⟨/natbib⟩

6.11 Output functions (continued)

`address.publisher` Outputs `address` (or `location`) and `publisher` fields separated by colon if both fields are available; otherwise outputs that field which is not empty.

```
⟨*!strict⟩
FUNCTION {output.address.publisher}
{
  address empty$ location empty$ and
    'skip$
    { address.or.location output
      publisher empty$
        'skip$
        { new.colon }
      if$
    }
  if$
  publisher output
}
⟨/!strict⟩
⟨*strict⟩
FUNCTION {output.address.publisher}
```

```
{
  address empty$
    {
      bbl.nnoaddress
      publisher empty$
        { "~: " * bbl.nopublisher * bracketise }
        { bracketise "~: " * publisher * }
      if$
    }
    {
      address output
      new.colon
      publisher empty$
        { bbl.nopublisher bracketise }
        { publisher }
      if$
    }
  if$
  output
}
```

⟨/strict⟩

`output.bibitem` Is called at the beginning of any entry. It sets `curlanguage` string variable based on `langid` or `language` field.

```
FUNCTION {output.bibitem}
{
  set.language
  newline$
  "\bibitem" write$
```

⟨∗natbib⟩

```
  label extra.label * ")" *
  calc.long.list *
  bracketise write$
```

⟨/natbib⟩

```
  cite$ bracify write$
  newline$
  "\selectlanguageifdefined" curlanguage bracify * write$
  newline$
  ""
  before.all 'output.state :=
}
```

⟨∗natbib⟩

```
%FUNCTION {output.bibitem}
%{ newline$
%  "\bibitem" write$
%%  author.key.label
%%  year parenthesify *
%%  "; lbl:" label * *
```

```
%%  "; mfn:"  calc.long.list * *
%  label
%  calc.long.list *
%  bracketise write$
%  cite$ bracify write$
%  newline$
%  language empty$
%    { "english" 'curlanguage := }
%    {language  'curlanguage := }
%  if$
%  "\selectlanguageifdefined" curlanguage bracify * write$
%  newline$
%  ""
%  before.all 'output.state :=
%}

⟨/natbib⟩
```

6.12 Formatting title, booktitle, etc.

Important note

Neither `bibtex.exe` nor `bibtex8.exe` 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 applied.

format.bvolume

```
FUNCTION {format.bvolume}
{ volume empty$
    { "" }
    { bbl.vvol volume tie.connect
      series empty$
        'skip$
        { bbl.of spaces.around * series emphasize * }
      if$
      "volume and number" number either.or.check
    }
  if$
}

```

rmat.number.series

```
FUNCTION {format.number.series}
{ volume empty$
    { number empty$
        { series field.or.null }
        { series empty$
            { "there's a number but no series in " cite$ * warning$
              bbl.nnr }
            {
              %new.dblslash
              new.sentence
```

```
            series
            bbl.nr
            tie.or.space.connect}
          if$
          number tie.or.space.connect
        }
      if$
    }
    { "" }
  if$
}

```

`eng.ord` Is not currently used. ⟨NB: Note that `bbl.st`, `bbl.nd`, `bbl.rd`, `bbl.th` are not defined.⟩

```
⟨*debug⟩
FUNCTION {eng.ord}
{ duplicate$ "1" swap$ *
  #-2 #1 substring$ "1" =
     { bbl.th * }
     { duplicate$ #-1 #1 substring$
       duplicate$ "1" =
         { pop$ bbl.st * }
         { duplicate$ "2" =
             { pop$ bbl.nd * }
             { "3" =
                 { bbl.rd * }
                 { bbl.th * }
               if$
             }
           if$
         }
       if$
     }
   if$
}
⟨/debug⟩

```

`convert.edition`

```
FUNCTION {convert.edition}
{ edition
% edition extract.num "l" change.case$ 's :=
%  s "first" = s "1" = or
%    { bbl.first 't := }
%    { s "second" = s "2" = or
%        { bbl.second 't := }
%        { s "third" = s "3" = or
%            { bbl.third 't := }
%            { s "fourth" = s "4" = or
%                { bbl.fourth 't := }
%                { s "fifth" = s "5" = or
```

```
%                    { bbl.fifth 't := }
%                    { s #1 #1 substring$ is.num
%                        { s eng.ord 't := }
%                        { edition 't := }
%                      if$
%                    }
%                  if$
%                }
%              if$
%            }
%          if$
%        }
%      if$
%    }
%  if$
%  t
}

format.edition

FUNCTION {format.edition}
{ edition empty$
    { "" }
    { output.state mid.sentence =
⟨!utf8⟩         { convert.edition "l" change.case$ " " * bbl.edition * }
⟨!utf8⟩         { convert.edition "t" change.case$ " " * bbl.edition * }
⟨utf8⟩         { convert.edition " " * bbl.edition * }
⟨utf8⟩         { convert.edition " " * bbl.edition * }
      if$
    }
  if$
}

format.pages

FUNCTION {format.pages}
{ eid empty$
    {
      pages empty$
        { "" }
        { pages multi.page.check
            { bbl.ppages pages n.dashify tie.connect }
            { bbl.ppage pages tie.connect }
          if$
        }
      if$
    }
    { eid multi.page.check
        { bbl.ppages eid n.dashify tie.connect }
        { bbl.ppage eid tie.connect }
      if$
```

```
    }
  if$
}

format.pages.page
FUNCTION {format.pages.page}
{ eid empty$
    { pages empty$
        { pagetotal empty$
            { "" }
            { pagetotal bbl.pages tie.connect }
          if$
        }
        { format.pages}
      if$
    }
    { format.pages }
  if$
}

rmat.vol.num.pages
FUNCTION {format.vol.num.pages}
{ volume field.or.null
  number empty$
    'skip$
    {
      ", no." number tie.or.space.connect *
      volume empty$
        { "there's a number but no volume in " cite$ * warning$ }
        'skip$
      if$
    }
  if$
  pages empty$
    'skip$
    { duplicate$ empty$
        { pop$ format.pages }
        { ": " * pages n.dashify * }
      if$
    }
  if$
}

  format.volume
FUNCTION {format.volume}
{ volume empty$
    { "" }
    { bbl.vvol volume tie.or.space.connect }
  if$
```

```
}

format.number
FUNCTION {format.number}
{ number empty$
    { "" }
    { bbl.nr number tie.or.space.connect }
  if$
}

rmat.chapter.pages
⟨*debug⟩
FUNCTION {format.chapter.pages}
{ chapter empty$
    'format.pages
    { type empty$
        { bbl.chapter }
        { type "l" change.case$ }
      if$
      chapter tie.or.space.connect
      pages empty$
        'skip$
        { ", " * format.pages * }
      if$
    }
  if$
}
⟨/debug⟩

empty.misc.check
FUNCTION {empty.misc.check}
{ author empty$ title empty$ howpublished empty$
  month empty$ year empty$ note empty$
  and and and and and
  key empty$ not and
    { "all relevant fields are empty in " cite$ * warning$ }
    'skip$
  if$
}

bbl.thesis.type
```

⟨NB: ЭТУ ФУНКЦИЮ ПЕРЕНЕСТИ КУДА-ТО ВНИЗ ближе к `format.thesis.type` И объединить с ней (?)⟩

```
FUNCTION {bbl.thesis.type}
{ type "mathesis" =
  { bbl.mathesis }
  { type "phdthesis" =
    { bbl.phdthesis }
```

```
    { type "docthesis" =
      { bbl.docthesis }
      %%{ "!!!" type * "t" change.case$ }
⟨!utf8⟩       { type "t" change.case$ }
⟨utf8⟩       { type }
    if$}
  if$}
if$}

```

ormat.thesis.type

```
%FUNCTION {format.thesis.type}
%{ type empty$
%    'skip$
%    { pop$
%%<!utf8>      bbl.thesis.type "t" change.case$
%%<utf8>      bbl.thesis.type
%    }
%  if$
%}
FUNCTION {format.thesis.type}
{ type empty$
    'skip$
    { pop$
      bbl.thesis.type
    }
  if$
}
```

echrep.type.number Function to format report type and number.

```
%FUNCTION {format.techrep.type.number}
%{ type empty$
%    { bbl.techreport }
%    'type
%  if$
%  number empty$
%%<!utf8>    { "t" change.case$ }
%%<utf8>    { "" }
%    { number tie.or.space.connect }
%  if$
%}

FUNCTION {format.techreport.type}
{ type empty$
   { bbl.techreport }
   'type
  if$
}

```

utput.author.head Formats and writes list of authors in the heading of a bibliographic record. The GOST bibstyles skip the list of authors in the beginning of the bibliographic record if the number of authors is 4

or larger except for the styles with suffix `l` which prints all authors in the heading. The bibstyles compiled with the option `struct` print at most 1 person name in the heading.

```
FUNCTION {output.author.head}
{
  author empty$
    'skip$
⟨*long⟩
    { author fmt.names.all output.nonnull
      %new.sentence
    }
⟨/long⟩
⟨*!long⟩
    {author num.names$ #4 <
      {
⟨strict⟩       author fmt.names.first output
⟨!strict⟩       author fmt.names.three output
       new.sentence
      }
      'skip$
    if$}
⟨/!long⟩
  if$
}

```

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

`utput.author.rest` Writes the rest of authors list after the entry title followed by a slash. In modern bibstyles, the list of authors in the heading part of a bibliographic record is cut to either 3 persons (if no `!strict` option is applied) or 1 person (if `strict` option). By idea, `output.author.rest` should print the rest of the authors in the `author` field but currently it also repeats the persons printed in the heading part. `output.author.rest` returns a null string (`""`) if all persons are printed in the heading part of the record. Note that all persons are alwats printed in the heading part if `long` option is applied.

```
FUNCTION {output.author.rest}
{
⟨*!long⟩
  author empty$
    'skip$
⟨strict⟩    {author num.names$ #1 >
⟨!strict⟩    {author num.names$ #3 >
      { author fmt.names.all output
        new.semicolon
      }
      'skip$
    if$}
  if$
⟨/!long⟩
}

```

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

```
FUNCTION {bookauthor.rest}
{
  bookauthor empty$
    'skip$
    {
⟨strict⟩       bookauthor fmt.names.all output
⟨!strict⟩      bookauthor fmt.names.three output
      new.semicolon
    }
  if$
}
```

organization.rest

```
FUNCTION {editor.organization.rest}
{
  compiler empty$
    {}
    { format.compiler.rest  output
    new.semicolon
    }
  if$
  editor empty$
    {}
    { format.editors.rest  output.nonnull
    new.semicolon
    }
  if$
  organization empty$
    {}
    {organization output.nonnull
    new.semicolon
    }
  if$
}
```

format.url

```
FUNCTION {format.url}
{ url empty$
    { "" }
    {
      bbl.url ": \BibUrl{" * url * "}" *
      urldate empty$
        { "" }
        { " (" bbl.urldate * ": " * urldate * ")" * }
      if$ *
    }
  if$
```

```
}

```

output.url

```
FUNCTION {output.url}
{
  url empty$
    'skip$
    { format.url output }
  if$
}

```

format.annote

```
FUNCTION {format.annote}
{ annote empty$
    { "" }
{ after.sentence 'output.state :=
  "\BibAnnote{" annote add.period$ * "}" *
}
  if$
}

```

format.isbn ⟨NB: Do we really need to provide electronic search for ISBN?⟩

```
FUNCTION {format.isbn}
{
  isbn empty$
    { "" }
    { "ISBN:~\href{http://isbndb.com/search-all.html?kw=" isbn *
      "}{" * isbn * "}" *
    }
  if$
}

```

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

Function `add.doi` embraces last string in stack into hyperlink that links it to specified `doi` identificator at https://doi.org/ web-site.

⟨∗eprint⟩

```
FUNCTION {add.doi}
{ duplicate$ empty$
    'skip$
    { doi empty$
        'skip$
        { "\href{https://doi.org/" doi * "}{" * swap$ * "}" * }
      if$
```

```
    }
  if$
}
```
⟨/eprint⟩

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

⟨∗!eprint⟩
```
FUNCTION {add.doi} { }
```
⟨/!eprint⟩

`add.media` New in version 1.2. Adds `media` field if `strict` options is in effect. If the `media` field is empty `add.media` prints a value based on the `type` field. If the `type` is also empty `add.media` prints equivalent of the word `text` in current language.

⟨∗!strict⟩
```
FUNCTION {add.media}
{ duplicate$ empty$
    'skip$
    { media empty$
        'skip$
        { " " * bbl.media bracketise * }
      if$
    }
  if$
}
```
⟨/!strict⟩
⟨∗strict⟩
```
FUNCTION {add.media}
{ duplicate$ empty$
    'skip$
    { media empty$
      { type$ "online" =
          { " " * bbl.media.online bracketise * }
          { " " * bbl.media.text   bracketise * }
        if$
      }
      { " " * bbl.media bracketise * }
    if$
    }
  if$
}
```
⟨/strict⟩

6.13 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.

⟨NB: This function should use url. TO BE DONE YET.⟩

```
⟨*eprint⟩
%FUNCTION {format.eprint}
%{ eprint empty$
%   { "" }
%   { eprintclass empty$
%      { " \href{http://arxiv.org/abs/" eprint * "}" * "{" * eprint * "}" * }
%      { eprinttype empty$
%         { " \href{http://arxiv.org/abs/" eprint * "}" *
%          "{" * eprintclass * "/" * eprint * "}" *
%         }
%         { " \href{http://arxiv.org/abs/" eprint * "}" *
%          "{" * eprinttype * ":" * eprintclass * "/" * eprint * "}" *
%         }
%      if$}
%   if$}
%if$}

%FUNCTION {format.eprint}
%{ eprint empty$
%   { "" }
%   { eprinttype empty$
%       { "" }
%       { eprinttype "~: " *}
%     if$
%     eprintclass empty$
%       { }
%       { eprintclass * "/" *}
%    if$
%    eprint *
%   }
%  if$
%  url empty$
%    { }
%    { "\href{" url * "}{" * swap$ * "}" *}
%  if$
%}

FUNCTION {format.eprint}
{ eprint empty$
```

```
    { "" }
    { eprinttype empty$
        { "" }
        { eprinttype "~: " *}
      if$
      eprintclass empty$
        { }
        { eprintclass * "/" *}
      if$
      url empty$
       { eprint * }
       { "\href{" * url * "}{" * eprint * "}" *}
     if$
    }
   if$
}

FUNCTION {output.eprint.url}
{
  eprint empty$
    { format.url output }
    { format.eprint output }
  if$
}
```

⟨/eprint⟩

⟨∗!eprint⟩

```
FUNCTION {output.eprint.url}
{
    format.url output
}
```

⟨/!eprint⟩

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

add.number

```
FUNCTION {add.number}
{ duplicate$ empty$
    { "" }
    { bbl.nr swap$ tie.or.space.connect }
  if$
}
```

ormat.type.number

```
FUNCTION {format.type.number}
{
  type empty$
    { "" }
    {
```

```
      number empty$
      { "" }
      { type number tie.or.space.connect }
    if$
    }
  if$
}

ormat.requestdate
FUNCTION {format.requestdate}
{ requestdate empty$
    { "" }
    { bbl.req requestdate tie.or.space.connect }
  if$
}

t.publicationdate
FUNCTION {format.publicationdate}
{ publicationdate empty$
    { "" }
    { bbl.publ publicationdate tie.or.space.connect }
  if$
}

rmat.prioritydate
FUNCTION {format.prioritydate}
{ prioritydate empty$
    { "" }
    { bbl.priority prioritydate tie.or.space.connect }
  if$
}
```

6.14 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.

6.14.1 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.

```
FUNCTION {article}
{
  output.bibitem
  output.author.head
  new.sentence
⟨natbib⟩  author format.key output
  title add.media "title" output.check
  new.slash
  output.author.rest
  new.dblslash
  journal emphasize add.doi "journal" output.check % new in v1.2
  new.block
  format.date "year/date" output.check
  new.block
  format.volume output
  format.number output
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`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.

```
FUNCTION {book}
{
```

```
  output.bibitem
  output.author.head
  new.sentence
⟨natbib⟩  author format.key output
  title add.doi add.media "title" output.check
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  new.slash
  output.author.rest
  editor.organization.rest
  new.sentence
  format.number.series output
  new.block
  format.edition output
  new.block
  output.address.publisher
  format.date "year/date" output.check
  new.block
  format.bvolume output
  new.block
  format.pages.page output
  new.block
⟨eprint⟩  format.isbn output
⟨eprint⟩  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

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.

```
FUNCTION {booklet}
{
  output.bibitem
  output.author.head
  new.sentence
⟨natbib⟩  author format.key output
  title add.doi add.media "title" output.check
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  new.slash
  output.author.rest
```

```
  editor.organization.rest
  new.block
  howpublished output
  address.or.location output
  format.date "year/date" output.check
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
```

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.

```
FUNCTION {inbook}
{
  output.bibitem
  output.author.head
  new.sentence
⟨natbib⟩  author format.key output
  title add.doi add.media "title" output.check
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  new.slash
  output.author.rest
  new.dblslash
%  bookauthor.head
  booktitle "booktitle" output.check
  new.slash
  bookauthor.rest
  editor.organization.rest
  new.block
  format.edition output
  new.block
  format.number.series output
  new.sentence
  output.address.publisher
  format.date "year/date" output.check
  new.block
  format.bvolume output
  new.block
```

```
  format.pages.page output
  new.block
⟨eprint⟩  format.isbn output
⟨eprint⟩  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
```

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.

```
FUNCTION {incollection}
{
  output.bibitem
  output.author.head
  new.sentence
⟨natbib⟩  author format.key output
  new.sentence
  title add.doi add.media "title" output.check
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  new.slash
  output.author.rest
  new.dblslash
  booktitle "booktitle" output.check
  new.slash
  editor.organization.rest
  new.block
  output.address.publisher
  format.date "year/date" output.check
  new.block
  format.bvolume output
  format.number.series output
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
```

```
  format.annote output
  fin.entry
}
```

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.

```
FUNCTION {proceedings}
{
  output.bibitem
⟨natbib⟩  editor format.key output
  title add.doi add.media "title" output.check
  new.colon                   % added in v.1.2k
  titleaddon output           % added in v.1.2k
  new.slash
  editor.organization.rest
  new.block
  output.address.publisher    % 1.2k moved from below
  new.block                   % added in v.1.2k
  format.date "year/date" output.check
  new.block
  format.bvolume output
  format.number.series output
  new.block
  format.pages.page output
  %%%output.address.publisher  % 1.2k moved upper
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
```

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.

```
FUNCTION {inproceedings}
{ output.bibitem
```

```
  output.author.head
  new.sentence
⟨natbib⟩  author format.key output
  new.sentence
  title add.doi add.media "title" output.check
  new.colon                % added in v.1.2k
  titleaddon output        % added in v.1.2k
  new.slash
  output.author.rest
  new.dblslash
  booktitle "booktitle" output.check
  new.slash
  editor.organization.rest
  new.block
  output.address.publisher    % 1.2k moved from below
  new.block                   % added in v.1.2k
%  address empty$
%    { publisher output
%      format.date "year/date" output.check
%    }
%    { address output.nonnull
%      new.colon
%      publisher output
%      format.date "year/date" output.check
%    }
%  if$
%  output.address.publisher        % moved upper in v.1.2k
  format.date "year/date" output.check
  new.block
  format.bvolume output
  format.number.series output
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`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.

```
FUNCTION {manual}
{ output.bibitem
```

```
  author empty$
    { organization empty$
        'skip$
        { organization output.nonnull
          address output
        }
      if$
    }
    { format.author output.nonnull }
  if$
⟨natbib⟩  author format.key output
  new.block
  title add.doi add.media "title" output.check
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  author empty$
    { organization empty$
    {
          address new.block.checka
          address output
        }
        'skip$
      if$
    }
    {
      organization address new.block.checkb
      organization output
      address output
    }
  if$
  format.edition output
  format.date "year/date" output.check
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`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.

```
FUNCTION {patent}
```

```
{
  output.bibitem
  title add.media output.nonnull
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  new.colon
  format.type.number output
  add.blank
  address.or.location output
  new.colon
  ipc output
  new.slash
  format.author "author" output.check
  add.blank
  authorcountry paranthesify output.nonnull
⟨natbib⟩  author format.key output
  new.semicolon
  holder output.nonnull
  new.semicolon
  credits output.nonnull
  new.block
  requestnumber add.number output
  new.semicolon
  format.requestdate output
  new.semicolon
  format.publicationdate output
  publication output
  new.semicolon
  format.prioritydate output
  prioritynumber output
  prioritycountry paranthesify output
  new.block
  note output
  new.sentence
  output.url
  format.annote output
  new.block
  pagetotal output
  fin.entry
}

```

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.

```
FUNCTION {misc}
{ output.bibitem
  %format.author output  % < v.1.2k
  output.author.head     % v.1.2k
```

```
  new.sentence           % v.1.2k
⟨natbib⟩  author format.key output
  title howpublished new.sentence.checkb
  title add.media output
  howpublished new.block.checka
  howpublished output
  new.block
  format.date "year/date" output.check
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`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

```
FUNCTION {unpublished}
{ output.bibitem
  output.author.head
  new.sentence
⟨natbib⟩  author format.key output
  title "title" output.check
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  new.slash
  output.author.rest
  editor.organization.rest
  new.block
  format.date "year/date" output.check
  new.block
  note "note" output.check
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`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.

```
FUNCTION {online}
{ output.bibitem
  format.author output
⟨natbib⟩  author format.key output
  title howpublished new.sentence.checkb
  title add.doi add.media "title" output.check
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
%  howpublished new.block.checka
  howpublished new.dblslash.checka
⟨!strict⟩  howpublished output
⟨strict⟩  howpublished bracketise output
  editor.organization.rest
  new.sentence
  new.block
  output.address.publisher
  format.date output
  new.block
%  format.url output
  output.eprint.url
  new.sentence
  note output
  format.annote output
  fin.entry
}

```

internet New in version 2012.02.15.

```
FUNCTION {internet}   {online}
FUNCTION {www}        {online}
FUNCTION {webpage}    {online}
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

```
FUNCTION {thesis}
{ output.bibitem
  format.author "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  title add.doi add.media "title" output.check
  new.colon
  bbl.phdthesis format.thesis.type output.nonnull
```

```
  new.colon
  specialitycode.or.number output % code of the speciality, new in v.1.2i
  new.colon
  titleaddon output % date of defence and approvement; new in v.1.2i
  new.slash
  %%%format.author.rest output %% dupplicates athours from the head zone
  output.author.rest            %% prints if num.names$ > 3 or > 1
  new.semicolon
  %institution "institution" output.check
  institution.or.school "institution/school" output.check
  new.block
  output.address.publisher
  format.date "year/date" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
```

`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.

```
%FUNCTION {report}
%{
%  output.bibitem
%  output.author.head
%  new.sentence
%  title add.doi add.media "title" output.check
%  new.colon
%%  format.techrep.type.number output.nonnull
%  type "type" output.check
%  new.slash
%  output.author.rest
%  editor.organization.rest
%  new.block
%  address output
%  new.colon
%  institution "institution" output.check
%  format.date "year/date" output.check
%  new.block
%  note output
```

```
%  new.block                % v.2
%  format.pages.page output % v.2
%  new.sentence
%%  format.url output
%  output.eprint.url
%  format.annote output
%  fin.entry
%}
FUNCTION {report}
{
  output.bibitem
%  output.author.head
%  new.sentence
  title add.doi add.media "title" output.check
⟨natbib⟩  title format.key output
  new.colon          % added in v.1.2k
  titleaddon output  % added in v.1.2k
  new.colon
%  format.techrep.type.number output.nonnull
%  type "type" output.check
%  format.report.type.number "type" output.check
  type "type" output.check
  new.colon
  number output
  new.slash
  %institution "institution" output.check
  institution.or.school "institution/school" output.check
  new.semicolon
  format.chief.rest output % from editor field
  new.semicolon
  format.executor.rest output % from author field
  new.block
  address.or.location output
  new.colon
  organization output
  format.date "year/date" output.check
  new.block                % v.2
  format.pages.page output % v.2
  new.sentence % или new.block ?
  output.eprint.url
  new.block
  note output
  format.annote output
  fin.entry
}

```

6.14.2 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.

```
FUNCTION {phdthesis}
{ output.bibitem
  format.author "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  title add.doi add.media "title" output.check
  new.colon
  bbl.phdthesis format.thesis.type output.nonnull
  new.colon
  %%number output % code of the speciality
  specialitycode.or.number output % code of the speciality, new in v.1.2i
  new.colon
  titleaddon output % date of defence and approvement; new in v.1.2i
  new.slash
  %%%format.author.rest output %% dupplicates athours
  output.author.rest            %% prints if num.names$ > 3 or > 1
  new.semicolon
  %institution "institution" output.check
  institution.or.school "institution/school" output.check
  new.block
  output.address.publisher
  format.date "year/date" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
```

`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.

```
FUNCTION {mastersthesis}
{ output.bibitem
  format.author "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  title add.doi add.media "title" output.check
  new.colon
```

```
  bbl.mathesis format.thesis.type output.nonnull
  new.colon
  %%number output % code of the speciality
  specialitycode.or.number output % code of the speciality, new in v.1.2i
  new.colon
  titleaddon output % date of defence and approvement; new in v.1.2i
  new.slash
  %%%format.author.rest output %% dupplicates athours
  output.author.rest            %% prints if num.names$ > 3 or > 1
  new.semicolon
  %institution "institution" output.check
  institution.or.school "institution/school" output.check
  new.block
  output.address.publisher
  format.date "year/date" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
```

`docthesis` 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.

```
FUNCTION {docthesis}
{ output.bibitem
  format.author "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  title add.doi add.media "title" output.check
  new.colon
  bbl.docthesis format.thesis.type output.nonnull
  new.colon
  %%number output % code of the speciality
  specialitycode.or.number output % code of the speciality, new in v.1.2i
  new.colon
  titleaddon output % date of defence and approvement; new in v.1.2i
  new.slash
  %%%format.author.rest output %% dupplicate athours
  output.author.rest            %% prints if num.names$ > 3 or > 1
  new.semicolon
  %institution "institution" output.check
  institution.or.school "institution/school" output.check
  new.block
  output.address.publisher
  format.date "year/date" output.check
  new.block
```

```
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
```

conference

```
FUNCTION {conference} { inproceedings }

```

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.

```
%FUNCTION {techreport}
%{
%  output.bibitem
%  output.author.head
%  new.sentence
%  title add.doi add.media "title" output.check
%  new.colon
%  format.techrep.type.number output.nonnull
%  new.slash
%  output.author.rest
%  editor.organization.rest
%  new.block
%  address output
%  new.colon
%  institution "institution" output.check
%  format.date "year/date" output.check
%  new.block
%  note output
%  new.block                % v.2
%  format.pages.page output % v.2
%  new.sentence
%%  format.url output
%  output.eprint.url
%  format.annote output
%  fin.entry
%}

FUNCTION {techreport}
{
  output.bibitem
%  output.author.head
%  new.sentence
  title add.doi add.media "title" output.check
⟨natbib⟩  title format.key output
  new.colon
```

```
3077 %  format.techrep.type.number output.nonnull
3078 %  type "type" output.check
3079 %  format.report.type.number "type" output.check
3080 %  type output
3081   format.techreport.type output
3082   new.colon
3083   number output
3084   new.slash
3085   %institution "institution" output.check
3086   institution.or.school "institution/school" output.check
3087   new.semicolon
3088   format.chief.rest output % from editor field
3089   new.semicolon
3090   format.executor.rest output % from author field
3091   new.block
3092   address.or.location output
3093   new.colon
3094   organization output
3095   format.date "year/date" output.check
3096   new.block                % v.2
3097   format.pages.page output % v.2
3098   new.sentence % или new.block ?
3099   output.eprint.url
3100   new.block
3101   note output
3102   format.annote output
3103   fin.entry
3104 }
3105
```

default.type

```
3106 FUNCTION {default.type} { misc }
3107
```

6.15 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.

```
3108 MACRO {jan} {"Jan."}
3109 MACRO {feb} {"Feb."}
3110 MACRO {mar} {"Mar."}
3111 MACRO {apr} {"Apr."}
3112 MACRO {may} {"May"}
3113 MACRO {jun} {"Jun."}
3114 MACRO {jul} {"Jul."}
3115 MACRO {aug} {"Aug."}
3116 MACRO {sep} {"Sep."}
3117 MACRO {oct} {"Oct."}
3118 MACRO {nov} {"Nov."}
```

```
MACRO {dec} {"Dec."}
```

6.16 Journal Abbreviations

6.16.1 Physics and astronomy

Borrowed from `physjour.mbs` of package custom-bib.

```
MACRO {aa}{"Astron.\ \& Astrophys."}
MACRO {aasup}{"Astron.\ \& Astrophys.\ Suppl.\ Ser."}
MACRO {aj} {"Astron.\ J."}
MACRO {aph} {"Acta Phys."}
MACRO {advp} {"Adv.\ Phys."}
MACRO {ajp} {"Amer.\ J.\ Phys."}
MACRO {ajm} {"Amer.\ J.\ Math."}
MACRO {amsci} {"Amer.\ Sci."}
MACRO {anofd} {"Ann.\ Fluid Dyn."}
MACRO {am} {"Ann.\ Math."}
MACRO {ap} {"Ann.\ Phys.\ (NY)"}
MACRO {adp} {"Ann.\ Phys.\ (Leipzig)"}
MACRO {ao} {"Appl.\ Opt."}
MACRO {apl} {"Appl.\ Phys.\ Lett."}
MACRO {app} {"Astroparticle Phys."}
MACRO {apj} {"Astrophys.\ J."}
MACRO {apjsup} {"Astrophys.\ J.\ Suppl."}
MACRO {apss} {"Astrophys.\ Space Sci."}
MACRO {araa} {"Ann.\ Rev.\ Astron.\ Astrophys."}
MACRO {baas} {"Bull.\ Amer.\ Astron.\ Soc."}
MACRO {baps} {"Bull.\ Amer.\ Phys.\ Soc."}
MACRO {cmp} {"Comm.\ Math.\ Phys."}
MACRO {cpam} {"Commun.\ Pure Appl.\ Math."}
MACRO {cppcf} {"Comm.\ Plasma Phys.\ \& Controlled Fusion"}
MACRO {cpc} {"Comp.\ Phys.\ Comm."}
MACRO {cqg} {"Class.\ Quant.\ Grav."}
MACRO {cra} {"C.\ R.\ Acad.\ Sci.\ A"}
MACRO {fed} {"Fusion Eng.\ \& Design"}
MACRO {ft} {"Fusion Tech."}
MACRO {grg} {"Gen.\ Relativ.\ Gravit."}
MACRO {ieeens} {"IEEE Trans.\ Nucl.\ Sci."}
MACRO {ieeeps} {"IEEE Trans.\ Plasma Sci."}
MACRO {ijimw} {"Interntl.\ J.\ Infrared \& Millimeter Waves"}
MACRO {ip} {"Infrared Phys."}
MACRO {irp} {"Infrared Phys."}
MACRO {jap} {"J.\ Appl.\ Phys."}
MACRO {jasa} {"J.\ Acoust.\ Soc.\ America"}
MACRO {jcp} {"J.\ Comp.\ Phys."}
MACRO {jchp} {"J.\ Chem.\ Phys."}
MACRO {jetp} {"Sov.\ Phys.--JETP"}
MACRO {jfe} {"J.\ Fusion Energy"}
MACRO {jfm} {"J.\ Fluid Mech."}
MACRO {jmp} {"J.\ Math.\ Phys."}
```

```
MACRO {jne} {"J.\ Nucl.\ Energy"}
MACRO {jnec} {"J.\ Nucl.\ Energy, C: Plasma Phys., Accelerators, Thermonucl.\ Res."}
MACRO {jnm} {"J.\ Nucl.\ Mat."}
MACRO {jpc} {"J.\ Phys.\ Chem."}
MACRO {jpp} {"J.\ Plasma Phys."}
MACRO {jpsj} {"J.\ Phys.\ Soc.\ Japan"}
MACRO {jsi} {"J.\ Sci.\ Instrum."}
MACRO {jvst} {"J.\ Vac.\ Sci.\ \& Tech."}
MACRO {nat} {"Nature"}
MACRO {nature} {"Nature"}
MACRO {nedf} {"Nucl.\ Eng.\ \& Design/Fusion"}
MACRO {nf} {"Nucl.\ Fusion"}
MACRO {nim} {"Nucl.\ Inst.\ \& Meth."}
MACRO {nimpr} {"Nucl.\ Inst.\ \& Meth.\ in Phys.\ Res."}
MACRO {np} {"Nucl.\ Phys."}
MACRO {npb} {"Nucl.\ Phys.\ B"}
MACRO {nt/f} {"Nucl.\ Tech./Fusion"}
MACRO {npbpc} {"Nucl.\ Phys.\ B (Proc.\ Suppl.)"}
MACRO {inc} {"Nuovo Cimento"}
MACRO {nc} {"Nuovo Cimento"}
MACRO {pf} {"Phys.\ Fluids"}
MACRO {pfa} {"Phys.\ Fluids A: Fluid Dyn."}
MACRO {pfb} {"Phys.\ Fluids B: Plasma Phys."}
MACRO {pl} {"Phys.\ Lett."}
MACRO {pla} {"Phys.\ Lett.\ A"}
MACRO {plb} {"Phys.\ Lett.\ B"}
MACRO {prep} {"Phys.\ Rep."}
MACRO {pnas} {"Proc.\ Nat.\ Acad.\ Sci.\ USA"}
MACRO {pp} {"Phys.\ Plasmas"}
MACRO {pop} {"Phys.\ Plasmas"}
MACRO {ppcf} {"Plasma Phys.\ \& Controlled Fusion"}
MACRO {phitrsl} {"Philos.\ Trans.\ Roy.\ Soc.\ London"}
MACRO {prl} {"Phys.\ Rev.\ Lett."}
MACRO {pr} {"Phys.\ Rev."}
MACRO {physrev} {"Phys.\ Rev."}
MACRO {pra} {"Phys.\ Rev.\ A"}
MACRO {prb} {"Phys.\ Rev.\ B"}
MACRO {prc} {"Phys.\ Rev.\ C"}
MACRO {prd} {"Phys.\ Rev.\ D"}
MACRO {pre} {"Phys.\ Rev.\ E"}
MACRO {ps} {"Phys.\ Scripta"}
MACRO {procrsl} {"Proc.\ Roy.\ Soc.\ London"}
MACRO {rmp} {"Rev.\ Mod.\ Phys."}
MACRO {rsi} {"Rev.\ Sci.\ Inst."}
MACRO {science} {"Science"}
MACRO {sciam} {"Sci.\ Am."}
MACRO {sam} {"Stud.\ Appl.\ Math."}
MACRO {st} {"Sky and Telesc."}
```

6.16.2 Supplementary Journal Names

Borrowed from `suppjour.mbs` of package custom-bib.

```
MACRO {cjp} {"Czech. J. Phys."}
MACRO {el} {"Europhys. Lett."}
MACRO {en} {"Europhys. News"}
MACRO {fujitsustj} {"FUJITSU Sci. Tech. J."}
MACRO {ieeeed} {"IEEE Trans. Electron Devices"}
MACRO {ieeeim} {"IEEE Trans. Instrum. Meas."}
MACRO {ieeejqe} {"IEEE J. Quantum Electron."}
MACRO {ieeem} {"IEEE Trans. Magn."}
MACRO {ieeeptl} {"IEEE Photonic Technol. Lett."}
MACRO {ieeeuffc} {"IEEE Trans. Ultrason., Ferroelect., Freq. Cont."}
MACRO {jem} {"J. Electron. Mater."}
MACRO {jes} {"J. Electrochem. Soc."}
MACRO {jetplett} {"JETP Lett."}
MACRO {jjap} {"Japan. J. Appl. Phys."}
MACRO {jpha} {"J. Phys. A: Math. Gen."}
MACRO {jphb} {"J. Phys. B: At. Mol. Opt. Phys."}
MACRO {jphbold} {"J. Phys. B: At. Mol. Phys."}
MACRO {jphc} {"J. Phys.: Condens. Matter"}
MACRO {jphcold} {"J. Phys. C: Solid State Phys."}
MACRO {jphd} {"J. Phys. D: Appl. Phys."}
MACRO {jvsta} {"J. Vac. Sci. Technol. A"}
MACRO {jvstb} {"J. Vac. Sci. Technol. B"}
MACRO {me} {"Microelectron. Eng."}
MACRO {necrd} {"NEC Res.{\&} Develop."}
MACRO {pa} {"Physica A"}
MACRO {pb} {"Physica B"}
MACRO {pc} {"Physica C"}
MACRO {pd} {"Physica D"}
MACRO {procieee} {"Proc. IEEE"}
MACRO {procspie} {"Proc. SPIE"}
MACRO {pssa} {"Phys. Stat. Sol. A"}
MACRO {pssb} {"Phys. Stat. Sol. B"}
MACRO {rpp} {"Rep. Progr. Phys."}
MACRO {sm} {"Synthet. Metal"}
MACRO {sost} {"Solid State Technol."}
MACRO {ss} {"Surf. Sci."}
MACRO {ssc} {"Solid State Commun."}
MACRO {sst} {"Semicond. Sci. Technol."}
MACRO {suplatt} {"Superlatt. Microstr."}
MACRO {sust} {"Supercond. Sci. Technol."}
MACRO {znat} {"Z. Naturforsch."}
```

6.16.3 Optics

Borrowed from `photjour.mbs`.

```
MACRO {appopt} {"Appl.\ Opt."}
MACRO {bell}   {"Bell Syst.\ Tech.\ J."}
```

```
MACRO {ell}    {"Electron.\ Lett."}
MACRO {jasp}   {"J.\ Appl.\ Spectr."}
MACRO {jqe}    {"IEEE J.\ Quantum Electron."}
MACRO {jlwt}   {"J.\ Lightwave Technol."}
MACRO {jmo}    {"J.\ Mod.\ Opt."}
MACRO {josa}   {"J.\ Opt.\ Soc.\ America"}
MACRO {josaa}  {"J.\ Opt.\ Soc.\ Amer.~A"}
MACRO {josab}  {"J.\ Opt.\ Soc.\ Amer.~B"}
MACRO {jdp}    {"J.\ Phys.\ (Paris)"}
MACRO {oc}     {"Opt.\ Commun."}
MACRO {ol}     {"Opt.\ Lett."}
MACRO {os}     {"Opt.\ Spectrosc."}
MACRO {phtl}   {"IEEE Photon. Technol. Lett."}
MACRO {pspie}  {"Proc.\ Soc.\ Photo-Opt.\ Instrum. Eng."}
MACRO {vr}     {"Vision Res."}
MACRO {zph}    {"Z.\ f.\ Physik"}
MACRO {zphb}   {"Z.\ f.\ Physik~B"}
MACRO {zphd}   {"Z.\ f.\ Physik~D"}
```

6.16.4 Physics of condensed Matter

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

6.16.5 Soviet and Russian journals

⟨№: To be extended.⟩

```
MACRO {sjpp} {"Sov.\ J.\ Plasma Phys."}
MACRO {spd}  {"Sov.\ Phys.--Doklady"}
MACRO {sptp} {"Sov.\ Phys.--Tech. Phys."}
MACRO {spu}  {"Sov.\ Phys.--Uspekhi"}
⟨!utf8⟩MACRO {ufn}  {"\CYRU\CYRF\CYRN"}
⟨utf8⟩MACRO {ufn}  {"УФН"}
MACRO {pu}  {"Phys.--Uspekhi"}
MACRO {sjot} {"Sov.\ J.\ Opt.\ Technol."}
MACRO {sjqe} {"Sov.\ J.\ Quantum Electron."}
MACRO {sleb} {"Sov.\ Phys.--Leb.\ Inst.\ Rep."}
MACRO {stph} {"Sov.\ Phys.--Techn.\ Phys."}
MACRO {stphl}{"Sov.\ Techn.\ Phys.\ Lett."}
```

6.17 Main cycle

```

READ

```

6.18 Sorting

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

⟨∗sort | natbib⟩

sortify

```
FUNCTION {sortify}
{ purify$
⟨!utf8⟩  "l" change.case$
}
⟨/sort | natbib⟩
```

sort.format.names

```
⟨*sort⟩
%% ====================================
%% This version from old Gost package
%%<*!natbib>
FUNCTION {sort.format.names}
{ 's :=
  #1 'nameptr :=
  ""
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
    { nameptr #1 >
        { "   " * }
        'skip$
      if$
      s nameptr
      "{vv{ } }{ll{ }}{  f{ }}{  jj{ }}"
      format.name$ 't :=
      nameptr numnames = t "others" = and
        { "et al" * }
        %{ bbl.etal * }
        { t sortify * }
      if$
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}
%%</!natbib>
%% This version from plainnat.bst
%% It ignores second and subsequent authors but include year.
%%<*natbib>
%FUNCTION {sort.format.names}
%{ 's :=
%  #1 'nameptr :=
%  ""
%  s num.names$ 'numnames :=
%  numnames 'namesleft :=
%    { namesleft #0 > }
%    {
%      s nameptr "{vv{ } }{ll{ }}{  ff{ }}{  jj{ }}" format.name$ 't :=
%      nameptr #1 >
%        {
```

```
3339 %          "   "  *
3340 %          namesleft #1 = t "others" = and
3341 %            { "zzzzz" * }
3342 %            { numnames #2 > nameptr #2 = and
3343 %                { "zz" * year field.or.null * "   " * }
3344 %                'skip$
3345 %              if$
3346 %              t sortify *
3347 %            }
3348 %          if$
3349 %        }
3350 %        { t sortify * }
3351 %      if$
3352 %      nameptr #1 + 'nameptr :=
3353 %      namesleft #1 - 'namesleft :=
3354 %    }
3355 %  while$
3356 %}
3357 %%</natbib>
3358 %% ====================================
3359
```

sort.format.title

```
3360 FUNCTION {sort.format.title}
3361 { 't :=
3362   "A " #2
3363     "An " #3
3364       "The " #4 t chop.word % Removes "The " if any
3365     chop.word               % Removes "An " if any
3366   chop.word                 % Removes "A " if any
3367   sortify
3368   #1 global.max$ substring$
3369 }
3370
```

author.sort

```
3371 %% ====================================
3372 %% This version from old gost package.
3373 %%
3374 ⟨∗!natbib⟩
3375 FUNCTION {author.sort}
3376 { author empty$
3377     { key empty$
3378         { "to sort, need author or key in " cite$ * warning$
3379           ""
3380         }
3381         { key sortify }
3382       if$
3383     }
3384     {
3385       author num.names$ #4 <
```

```
        {author sort.format.names }
        {title sort.format.title}
      if$
    }
  if$
}
⟨/!natbib⟩
%% This version from plainnat.bst
⟨∗natbib⟩
FUNCTION {author.sort}
{ author empty$
    { key empty$
        { "to sort, need author or key in " cite$ * warning$
          ""
        }
        { key sortify }
      if$
    }
    { author sort.format.names }
  if$
}
⟨/natbib⟩
%% ====================================
```

`author.title.sort` The function `author.title.sort` is used in the `presort` function only.

```
FUNCTION {author.title.sort}
{ author empty$
    { title empty$
        { key empty$
            { "to sort, need author, title, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { title sort.format.title }
      if$
    }
    {
      author num.names$ #4 <
        {author sort.format.names }
```

For styles compiled with the option `long`, the list of authors always precedes book or article title no matter how long is it. Hence the entries should be sorted by author names.

```
⟨long⟩        {author sort.format.names }
⟨!long⟩        {title sort.format.title}
      if$
     }
  if$
}
```

uthor.editor.sort
organization.sort
organization.sort

The function `author.editor.sort`, `author.organization.sort`, `editor.organization.sort` are not currently used. See commented text in function `presort`#2. ⟨NB: WE NEED TO CORRECT THAT!⟩

```
⟨*natbib⟩
%FUNCTION {author.editor.sort}
%{ author empty$
%    { editor empty$
%        { key empty$
%            { "to sort, need author, editor, or key in " cite$ * warning$
%              ""
%            }
%            { key sortify }
%          if$
%        }
%        { editor sort.format.names }
%      if$
%    }
%    { author sort.format.names }
%  if$
%}
%
%FUNCTION {author.organization.sort}
%{ author empty$
%    { organization empty$
%        { key empty$
%            { "to sort, need author, organization, or key in " cite$ * warning$
%              ""
%            }
%            { key sortify }
%          if$
%        }
%        { "The " #4 organization chop.word sortify }
%      if$
%    }
%    { author sort.format.names }
%  if$
%}

FUNCTION {editor.organization.sort}
{ editor empty$
    { organization empty$
        { key empty$
            { "to sort, need editor, organization, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { "The " #4 organization chop.word sortify }
      if$
```

```
    }
    { editor sort.format.names }
  if$
}
```
⟨/natbib⟩

presort Function to compute `sort.key$`. What is the space string "␣␣␣" for? Version #1 is for 'sort' and '!natbib' options. Version #2 is for 'sort' and 'natbib' options. Version #3 is for '!sort' and 'natbib' options.

```
⟨*!natbib⟩
FUNCTION {presort}%#1
{
  author.title.sort
  "   "
  *
  year field.or.null sortify
  *
  "   "
  *
  title field.or.null
  sort.format.title
  *
  #1 entry.max$ substring$
  'sort.key$ :=
}
⟨/!natbib⟩
⟨*natbib⟩
FUNCTION {presort}%#2
{ calc.label
  label sortify
  %author.title.sort
  "    "
  *
  % ========= plainnat.bst =========
%  type$ "book" =
%  type$ "inbook" =
%  or
%    'author.editor.sort
%    { type$ "proceedings" =
%        'editor.organization.sort
%        { type$ "manual" =
%            'author.organization.sort
%            'author.sort
%          if$
%        }
%      if$
%    }
%  if$
  author.title.sort
  "    "
  *
```

```
  year field.or.null sortify
  *
  "    "
  *
  %cite$
  title field.or.null sort.format.title
  *
  #1 entry.max$ substring$
  'sort.label :=
  sort.label *
  % ================================
  #1 entry.max$ substring$
  'sort.key$ :=
}
⟨/natbib⟩
⟨/sort⟩

⟨*!sort⟩
⟨*natbib⟩
INTEGERS { seq.num }

FUNCTION {init.seq}
{ #0 'seq.num :=}

EXECUTE {init.seq}

FUNCTION {int.to.fix}
{ "000000000" swap$ int.to.str$ *
  #-1 #10 substring$
}

FUNCTION {presort}%#3
{
  calc.label               % computes label
  label sortify            % initiates sort.label
  "    "
  *
  seq.num #1 + 'seq.num := % advance seq.num
  seq.num  int.to.fix      % prepend seq.num with 0s
  'sort.label :=           % set sort.label to seq.num
  sort.label *             % append seq.num to label
  #1 entry.max$ substring$ % cut if too long
  'sort.key$ :=            % set sort.key$
}
⟨/natbib⟩
⟨/!sort⟩

⟨*sort | natbib⟩
ITERATE {presort}

```

```
SORT

⟨/sort | natbib⟩

```

6.19 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.

```
⟨!natbib⟩STRINGS { longest.label }
⟨natbib⟩STRINGS { longest.label last.label next.extra }

⟨!natbib⟩INTEGERS { number.label longest.label.width }
⟨natbib⟩INTEGERS { number.label longest.label.width last.extra.num  }

```

ize.longest.label Initialize those string.

```
⟨*!natbib⟩
FUNCTION {initialize.longest.label}
{ "" 'longest.label :=
  #1 'number.label :=
  #0 'longest.label.width :=
}
⟨/!natbib⟩
⟨*natbib⟩
FUNCTION {initialize.longest.label}
{ "" 'longest.label :=
  #0 int.to.chr$ 'last.label :=
  "" 'next.extra :=
  #0 'longest.label.width :=
  #0 'last.extra.num :=
  #0 'number.label :=
}
⟨/natbib⟩

EXECUTE {initialize.longest.label}

```

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

```
⟨*!natbib⟩
FUNCTION {forward.pass}
{ number.label int.to.str$ 'label :=
  number.label #1 + 'number.label :=
  label width$ longest.label.width >
    { label 'longest.label :=
      label width$ 'longest.label.width :=
    }
    'skip$
```

```
  if$
}
⟨/!natbib⟩
⟨*natbib⟩
FUNCTION {forward.pass}
{ last.label label =
    { last.extra.num #1 + 'last.extra.num :=
      last.extra.num int.to.chr$ 'extra.label :=
    }
    { "a" chr.to.int$ 'last.extra.num :=
      "" 'extra.label :=
      label 'last.label :=
    }
  if$
  number.label #1 + 'number.label :=
}
⟨/natbib⟩

ITERATE {forward.pass}

```

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

```
⟨*natbib⟩
FUNCTION {reverse.pass}
{ next.extra "b" =
    { "a" 'extra.label := }
    'skip$
  if$
  extra.label 'next.extra :=
  extra.label
  duplicate$ empty$
    'skip$
    { "{\natexlab{" swap$ * "}}" * }
  if$
  'extra.label :=
  %%%label extra.label * 'label :=
}

REVERSE {reverse.pass}

FUNCTION {bib.sort.order}
{ sort.label  'sort.key$ :=
}

ITERATE {bib.sort.order}

SORT
⟨/natbib⟩

```

begin.bib Within thebibliography environment we define few formatting macros for user to customize how

the reference list is formatted.

```
FUNCTION {begin.bib}
{ "\begin{thebibliography}{"  longest.label  * "}" * write$ newline$
  "\def\selectlanguageifdefined#1{"                   write$ newline$
  "\expandafter\ifx\csname date#1\endcsname\relax"   write$ newline$
%  "\else\language\csname l@#1\endcsname\fi}"         write$ newline$
  "\else\selectlanguage{#1}\fi}"                      write$ newline$
  "\providecommand*{\href}[2]{{\small #2}}"           write$ newline$
  "\providecommand*{\url}[1]{{\small #1}}"            write$ newline$
  "\providecommand*{\BibUrl}[1]{\url{#1}}"            write$ newline$
  "\providecommand{\BibAnnote}[1]{}"                  write$ newline$
  "\providecommand*{\BibEmph}[1]{#1}"                 write$ newline$
  "\ProvideTextCommandDefault{\cyrdash}{\iflanguage{russian}{\hbox to.8em{--\hss--}}{\textemdash}}" write$ ne
  "\providecommand*{\BibDash}{\ifdim\lastskip>0pt\unskip\nobreak\hskip.2em plus 0.1em\fi" write$ newline$
  "\cyrdash\hskip.2em plus 0.1em\ignorespaces}" write$ newline$
  "\renewcommand{\newblock}{\ignorespaces}" write$ newline$
⟨natbib⟩  "\providecommand{\natexlab}[1]{#1}" write$ newline$
  preamble$ empty$
    'skip$
    { preamble$ write$ newline$ }
  if$
}


EXECUTE {begin.bib}

EXECUTE {init.state.consts}

ITERATE {call.type$}

end.bib

FUNCTION {end.bib}
{ newline$
%  "\catcode`\/=11"        write$ newline$
  "\end{thebibliography}" write$ newline$
}

EXECUTE {end.bib}

⟨/bst⟩
```

7 Change History

8 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.