Package 'RefManageR'

January 20, 2025

Version 1.4.0

Title Straightforward 'BibTeX' and 'BibLaTeX' Bibliography Management

Maintainer Mathew W. McLean <mathew.w.mclean@gmail.com>

Description Provides tools for importing and working with bibliographic references. It greatly enhances the 'bibentry' class by providing a class 'BibEntry' which stores 'BibTeX' and 'BibLaTeX' references, supports 'UTF-8' encoding, and can be easily searched by any field, by date ranges, and by various formats for name lists (author by last names, translator by full names, etc.). Entries can be updated, combined, sorted, printed in a number of styles, and exported. 'BibTeX' and 'BibLaTeX' '.bib' files can be read into 'R' and converted to 'BibEntry' objects. Interfaces to 'NCBI Entrez', 'CrossRef', and 'Zotero' are provided for importing references and references can be created from locally stored 'PDF' files using 'Poppler'. Includes functions for citing and generating a bibliography with hyperlinks for documents prepared with 'RMarkdown' or 'RHTML'.

License GPL-2 | GPL-3 | BSD_3_clause + file LICENSE

Imports xml2, jsonlite, utils, plyr, tools, httr, lubridate (>= 1.5.0), stringr, methods, bibtex (>= 0.4.1)

Suggests knitr, markdown, testthat

Encoding UTF-8

Depends R (>= 3.0)

VignetteBuilder knitr

BugReports https://github.com/ropensci/RefManageR/issues

URL https://github.com/ropensci/RefManageR/

RoxygenNote 7.2.1

NeedsCompilation no

Author Mathew W. McLean [aut, cre] (<https://orcid.org/0000-0002-7891-9645>)

Repository CRAN

Date/Publication 2022-09-30 22:40:08 UTC

Contents

`RefManageR-package` *Import and Manage BibTeX and BibLaTeX references with RefManageR*

Description

RefManageR provides tools for importing and working with bibliographic references. It greatly enhances the bibentry class by providing a class BibEntry which stores BibTeX and BibLaTeX references, supports UTF-8 encoding, and can be easily searched by any field, by date ranges, and by various formats for name lists (author by last names, translator by full names, etc.). Entries can be updated, combined, sorted, printed in a number of styles, and exported. BibTeX and BibLaTeX .bib files can be read into R and converted to BibEntry objects. Interfaces to NCBI's Entrez, CrossRef, and Zotero are provided for importing references and references can be created from locally stored PDFs using Poppler. Includes functions for citing and generating a bibliography with hyperlinks for documents prepared with RMarkdown or RHTML.

Details

Importing and Creating References

BibEntry objects can be created directly using the `BibEntry` function. `.bib` files can be read into R using the `ReadBib` function. Tools are provided for importing references from Crossref, Zotero, Google Scholar, and PDFs and looking up PubMed ID's and DOIs. See `ReadPDFs`, `ReadZotero`, `ReadCrossRef`, `ReadGS`, `ReadPubMed`, `GetPubMedByID`, `GetPubMedRelated`.

Manipulating BibEntry objects

BibEntry objects may be searched and indexed by field values, name lists, keys, dates, date ranges, etc. See `[.BibEntry`, `[<-.BibEntry`, `[[.BibEntry`, `$.BibEntry`.

Printing and Exporting Bibliographies

The `print.BibEntry` function can print in a number of formats (e.g. text, html) and most of the base bibliography styles available with BibLaTeX (e.g. alphabetic, numeric, authortitle, and authoryear). `toBibtex.BibEntry` will convert a BibEntry object to a character vector containing lines of a BibTeX file, converting fields, entry types and expanding crossreferences as needed to coerce BibLaTeX entries to BibTeX. `toBiblatex` converts the BibEntry object to a character vector containing lines of the corresponding BibLaTeX file. The results can be written to a file using `WriteBib`.

Citations can be generated in a number of styles using one of the available functions for citations. A list of references can be printed based on the works the user has cited thus far in their document. See `Cite`. The citations and bibliography can be printed including hyperlinks using either the R Markdown or R HTML formats.

Additional features

All sorting methods for bibliographies available in the BibLaTeX LaTeX package have been implemented see `sort.BibEntry` and the references.

Using `open.BibEntry` electronic copies of references can be opened in a PDF viewer or web browser.

The convenience function `BibOptions` is provided for setting defaults for commonly used functions such as `print.BibEntry`, `[.BibEntry`, and `Cite`. Its interface is similar to `options`.

Author(s)

McLean, M. W. <mathew.w.mclean@gmail.com>

References

McLean, M. W. (2014). Straightforward Bibliography Management in R Using the RefManageR Package. arXiv: 1403.2036 [cs.DL]. Submitted.

Kime, P., M. Wemheuer, and P. Lehman (2022). The biblatex Package. http://mirrors.ibiblio.org/CTAN/macros/latex/contrib/biblatex/doc/biblatex.pdf.

Hornik, K., D. Murdoch, and A. Zeileis (2012). Who Did What? The Roles of R Package Authors and How to Refer to Them. The R Journal **4**, 1. https://journal.r-project.org/archive/2012-1/RJournal_2012-1_Hornik~et~al.pdf

Patashnik, O (1988). Bibtexing. https://tug.org/texmf-docs/bibtex/btxdoc.pdf.

`+.BibEntry` *Merge two BibEntry objects while discarding duplicates*

Description

Merges two BibEntry objects comparing only the specified fields to detect duplicates, thus it is can be made less strict than using `duplicated`, `unique`, etc. Attributes are also merged and keys are ensured to be unique. `merge` and `+` simply provide different interfaces for merging.

Usage

```
## S3 method for class 'BibEntry'
e1 + e2

## S3 method for class 'BibEntry'
merge(
  x,
  y,
  fields.to.check = BibOptions()$merge.fields.to.check,
  ignore.case = BibOptions()$ignore.case,
  ...
)
```

Arguments

`e1`	BibEntry object
`e2`	BibEntry object to be merged with e1
`x`	BibEntry object
`y`	BibEntry object
`fields.to.check`	character vector; which BibLaTeX fields should be checked to determine if an entry is a duplicate? Can include `"bibtype"` to check entry type and `"key"` to check entry keys. Specifying `"all"` checks all fields using `duplicated`.
`ignore.case`	logical; if `TRUE`, case is ignored when determining if fields are duplicates.
`...`	ignored

Value

an object of class BibEntry

Author(s)

McLean, M. W. <mathew.w.mclean@gmail.com>

See Also

duplicated, unique

Other operators: $.BibEntry(), $<-.BibEntry(), [.BibEntry(), [<-.BibEntry(), [[.BibEntry(), [[<-.BibEntry(), c.BibEntry()

Examples

```
if (requireNamespace("bibtex")) {
    file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
    bib <- suppressMessages(ReadBib(file.name))
    bib1 <- bib[seq_len(44)]
    bib2 <- bib[45:length(bib)]

    ## The following is FALSE because the parent entry of one entry in bib1
    ##   is in bib2, so the child entry is expanded in the BibEntry object
    ##   returned by `[` to include the fields inherited from the dropped parent
    identical(merge(bib1, bib2, 'all'), bib)
    toBiblatex(bib1[[1L]])
    toBiblatex(bib[[1L]])

    ## Alternatively, the operator `[[` for BibEntry objects does not expand
    ##   cross references
    bib1 <- bib[[seq_len(44)]]
    bib2 <- bib[[45:length(bib)]]
    identical(merge(bib1, bib2, 'all'), bib)

    ## Not strict enough
    invisible(merge(bib1, bib2, c('title', 'date')))
 }

## New publications of R.J. Carroll from Google Scholar and Crossref
## Not run:
if (requireNamespace("bibtex")) {
    bib1 <- ReadGS(scholar.id = "CJOHNoQAAAAJ", limit = '10', sort.by.date = TRUE)
    bib2 <- ReadCrossRef(query = "rj carroll", limit = 10, sort = "relevance",
      min.relevance = 80)
    oldopt <- BibOptions(merge.fields.to.check = "title")
    rjc.new.pubs <- bib1 + bib2
    BibOptions(oldopt)
}

## End(Not run)
```

as.BibEntry *Coerce to a BibEntry object*

Description

Functions to check if an object is a BibEntry, or coerce it if possible.

Usage

```
as.BibEntry(x)

is.BibEntry(x)
```

Arguments

x	any R object.

Details

as.BibEntry is able to coerce suitably formatted character vectors, bibentry objects, lists, and data.frames to BibEntry objects. See the examples.

Value

as.BibEntry - if successful, an object of class BibEntry.

is.BibEntry - logical; TRUE if x is a BibEntry object.

Note

Each entry to be coerced should have a bibtype, key, and all required fields for the specified bibtype.

See Also

BibEntry

Examples

```
if (requireNamespace("bibtex")) {
    file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
    bib <- suppressMessages(ReadBib(file.name))[[20:21]]
    identical(as.BibEntry(unlist(bib)), bib)  ## see also RelistBibEntry

    identical(as.BibEntry(unclass(bib)), bib)

    identical(as.BibEntry(as.data.frame(bib)), bib)
 }

bib <- c(bibtype = "article", key = "mclean2014", title = "My New Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-01")
```

```
as.BibEntry(bib)

bib <- bibentry(bibtype = "article", key = "mclean2014", title = "My New Article",
journal = "The Journal", year = 2014, author = "Mathew W. McLean")
print(bib, .bibstyle = "JSS")
as.BibEntry(bib)

bib <- list(c(bibtype = "article", key = "mclean2014a", title = "My New Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-01"),
  c(bibtype = "article", key = "mclean2014b", title = "Newer Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-02"))
as.BibEntry(bib)
```

`as.data.frame.BibEntry` *Coerce to a Data Frame*

Description

Coerces a BibEntry object to a data.frame, with each row of the data frame being a field present in at least one entry in the BibEntry object being coerced.

Usage

```
## S3 method for class 'BibEntry'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	- a BibEntry object
`row.names`	- ignored
`optional`	- ignored
`...`	- ignored

Value

a data.frame object with row names giving the keys, and first column giving entry type.

See Also

`BibEntry`, `as.BibEntry`

Examples

```
bib <- list(c(bibtype = "article", key = "mclean2014a", title = "My New Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-01"),
  c(bibtype = "article", key = "mclean2014b", volume = 10, title = "My Newer Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-02"))
bib <- as.BibEntry(bib)
as.data.frame(bib)
```

BibEntry *Enhanced Bibliographic Entries*

Description

Provides a new class BibEntry which builds on bibentry to provide enhanced functionality for representing, manipulating, importing, etc. bibliographic information in BibTeX or BibLaTeX style.

Usage

```
BibEntry(
  bibtype,
  textVersion = NULL,
  header = NULL,
  footer = NULL,
  key = NULL,
  ...,
  other = list(),
  mheader = NULL,
  mfooter = NULL
)
```

Arguments

bibtype	a character string with a BibTeX entry type. See Entry Types for details.
textVersion	a character string with a text representation of the reference to optionally be employed for printing.
header	a character string with optional header text.
footer	a character string with optional footer text.
key	a character string giving the citation key for the entry.
...	arguments of the form tag = value giving the fields of the entry, with tag and value the name and value of the field, respectively. Arguments with empty values are dropped. See Entry Fields for details.
other	list; additional way to specify fields and their values
mheader	string; optional "outer" header text
mfooter	string; optional "outer" footer text

Details

The BibEntry objects created by BibEntry can represent an arbitrary positive number of references, as with bibentry, but many additional methods are defined for building and manipulating a database of references.

Value

an object of class BibEntry

Entry Types

bibentry creates "bibentry" objects, which are modeled after BibLaTeX and BibTeX entries. The entry should be a valid BibLaTeX or BibTeX entry type. For a list of valid BibTeX entry types, see `bibentry`. BibLaTeX supports all entry types from BibTeX for backwards compatibility. BibLaTeX defines following entry types '

- *article* - An article in a journal, magazine, newspaper, or other periodical which forms a self-contained unit with its own title. Required fields: author, title, journal/journaltitle, year/date.
- *book* - A single-volume book with one or more authors where the authors share credit for the work as a whole. Required fields: author, title, year/date. (Also covers BibTeX @inbook).
- *mvbook* - A multi-volume *book*. For backwards compatibility, multi-volume books are also supported by the entry type @book. Required fields: author, title, year/date.
- *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
- *bookinbook* This type is similar to *inbook* but intended for works originally published as a stand-alone book.
- *suppbook* - Supplemental material in a *book*. This type is closely related to the *inbook* entry type.
- *booklet* - A book-like work without a formal publisher or sponsoring institution. Required fields: author/editor, title, year/date.
- *collection* - A single-volume collection with multiple, self-contained contributions by distinct authors which have their own title. Required fields: editor, title, year/date.
- *mvcollection* - A multi-volume *collection*. Also supported by *collection*. Required fields: editor, title, year/date.
- *incollection* - A contribution to a collection which forms a self-contained unit with a distinct author and title. Required fields: author, editor, title, booktitle, year/date.
- *suppcollection* - Supplemental material in a *collection*.
- *manual* - Technical or other documentation, not necessarily in printed form. Required fields: author/editor, title, year/date
- *misc* - A fallback type for entries which do not fit into any other category. Required fields: author/editor, title, year/date
- *online* - An online resource. Required fields: author, title, number, year/date.
- *patent* - A patent or patent request. Required fields: author, title, number, year/date.
- *periodical* - A complete issue of a periodical, such as a special issue of a journal. Required fields: editor, title, year/date
- *suppperiodical* - Supplemental material in a *periodical*.
- *proceedings* - A single-volume conference proceedings. Required fields: editor, title, year/date.
- *mvproceedings* - A multi-volume @proceedings entry. Required fields: editor, title, year/date.

- *inproceedings* - An article in a conference proceedings. Required fields: author, editor, title, booktitle, year/date.
- *reference* - A single-volume work of reference such as an encyclopedia or a dictionary. Alias for *collection* in standard styles.
- *mvreference* - A multi-volume *reference* entry.
- *inreference* - An article in a work of reference. Alias for *incollection* in most styles.
- *report* - A technical report, research report, or white paper published by a university or some other institution. Required fields: author, title, type, institution, year/date.
- *set* - An entry set. This entry type is special, see BibLaTeX manual.
- *thesis* - 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.
- *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. Required fields: author, title, year/date.
- *xdata* - This entry type is special. *xdata* entries hold data which may be inherited by other entries. (Biber only.)
- *custom[a-f]* Custom types (up to five) for special bibliography styles. Not used by the standard styles.

Note

Date fields are parsed using the locale specified by `Sys.getlocale("LC_TIME")` (relevant when specifying a character 'month' field, instead of the recommended integer format)

Name list fields (author, editor, etc.) should be specified as they would be for BibTeX/BibLaTeX; e.g. `author = "Doe, Jane and Smith, Bob A."`.

Author(s)

McLean, M. W. <mathew.w.mclean@gmail.com>

References

BibLaTeX manual http://mirrors.ibiblio.org/CTAN/macros/latex/contrib/biblatex/doc/biblatex.pdf

See Also

bibentry

Examples

```
BibEntry(bibtype = "Article", key = "mclean2014", title = "An Article Title",
  author = "McLean, Mathew W. and Wand, Matt P.", journaltitle = "The Journal Title",
  date = "2014-02-06", pubstate = "forthcoming")
bib <- BibEntry(bibtype = "XData", key = "arxiv_data", eprinttype = "arxiv",
eprintclass = "stat.ME", year = 2013, urldate = "2014-02-01", pubstate = "submitted")
bib <- c(bib, BibEntry(bibtype = "Misc", key = "mclean2014b",
```

```
  title = "Something On the {arXiv}", author = "Mathew W. McLean", eprint = "1312.9999",
  xdata = "arxiv_data", url = "https://arxiv.org/abs/1310.5811"))
bib
toBiblatex(bib)
```

`BibOptions` *Set options/hooks for RefManageR*

Description

This function is used to access and set package options for RefManageR, similar to `options`. The options are listed in the details

Usage

```
BibOptions(..., restore.defaults = FALSE)
```

Arguments

`...`	a character vector or strings specifying option names to access; or to set options values, a named list or vector of option values or options specified in name=value pairs.
`restore.defaults`	logical; if TRUE, . . . 's are ignored and all package options are restored to their defaults.

Details

The following are valid package options.

Options for searching/indexing a BibEntry object. See `[.BibEntry` and `[<-.BibEntry`

1. `match.author` - string; controls how name list fields (author, editor, translator, etc.) are matched when searching for names. "family.with.initials" require family names and given name initials to match, "exact" requires names to match exactly, and any other value results in only family names being compared (the default).
2. `match.date` - string; controls how date fields are matched when searching. If "year.only" (the default), only years are checked for equality when comparing dates, otherwise months and days will also be compared, if they are available.
3. `use.regex` - logical; if `TRUE`, regular expressions are used when searching non-date fields; otherwise, exact matching is used.
4. `ignore.case` - logical; if `TRUE`, case is ignored when searching.
5. `return.ind` - logical; if `TRUE` the return value of `SearchBib` and the operators `[.BibEntry`, will be the indices of any matches; otherwise, a `BibEntry` object is returned.

Options for Printing with `print.BibEntry` and `PrintBibliography`

1. bib.style - string; Biblatex bibliography style to use when printing and formatting a BibEntry object. Possible values are "numeric" (default), "authoryear", "authortitle", "alphabetic", "draft".
2. first.inits - logical; if TRUE, only given name initials are displayed when printing; otherwise, full names are used.
3. dashed - logical; if TRUE and bib.style = "authoryear" or bib.style = "authortitle", recurring author and editor names are replaced with "—" when printing.
4. sorting - string; controls how BibEntry objects are sorted. Possible values are "nty", "nyt", "nyvt", "anyt", "anyvt", "ynt", "ydnt", "none", "debug"; see sort.BibEntry
5. max.names - numeric; maximum number of names to display before using "et al." when formatting and printing name list fields. This is also the minimum number of names that will be displayed if "et al." is used (corresponding to the 'minnames' package option in Biblatex). See below and option longnamesfirst when using this argument with the citation functions (Citet, etc.).
6. no.print.fields character vector; fields that should not be printed, e.g., doi, url, isbn, etc.
7. style - character string naming the printing style. Possible values are plain text (style "text"), BibTeX ("Bibtex"), BibLaTeX ("Biblatex"), a mixture of plain text and BibTeX as traditionally used for citations ("citation"), HTML ("html"), LaTeX ("latex"), "markdown", "yaml", R code ("R"), and a simple copy of the textVersion elements (style "textVersion", see BibEntry)

Options for the Cite functions

1. cite.style - character string; bibliography style to use to generate citations.
2. style - as above, but used to format the citations.
3. hyperlink - character string or logical; for use with style = "markdown" and style = "html" (ignored otherwise). If FALSE, no hyperlink will be generated for the citation or in the bibliography when printing. If set equal to "to.bib", then hyperlinks will be generated linking the citation and bibliography. The default value, "to.doc", will try to create the hyperlink using the url, doi, or eprint fields of entry. If these fields are not available, the hyperlink will point to the bibliography. See also open.BibEntry.
4. super - logical; should superscripts be used for numeric citations? Ignored if cite.style != "numeric".
5. max.names - numeric; same as above, except for citations. Note, that the first time a reference is cited, this option will be ignored if longfirstnames is TRUE.
6. longnamesfirst logical; should the first time a citation appears in the text not be truncated at max.names?
7. bibpunct - character vector; punctuation to use in a citation. The entries in bibpunct are as follows
   (a) The left delimiter for non-alphabetic and non-numeric citation styles
   (b) The right delimiter for non-alphabetic and non-numeric citation styles
   (c) The left delimiter for alphabetic and numeric citation styles
   (d) The right delimiter for alphabetic and numeric citation styles
   (e) The separator between references in a citation.
   (f) Punctuation to go between the author and year.

Other

1. `check.entries` - string or `FALSE`; if `FALSE` entries are not checked to ensure that they have all the required fields for the type of entry; if "warn" then entries are checked, but only a warning is issued and the entry is processed anyway; otherwise an error is produced if an entry does not have the required fields (default). Note that the majority of fields listed as required for a particular entry type in the Biblatex manual are not actually required for Biblatex to produce an entry.
2. `merge.fields.to.check` - character vector; for `merge.BibEntry` and the operator `+.BibEntry`, the fields that should be checked when comparing entries for equality when merging BibEntry objects. Specifying "all" results in all fields be checked with `duplicated`. The default is "key" to only check for duplicated keys.

Value

if a vector of option names is supplied, the current value of the requested options, or if `...` is missing, all current option values; otherwise, when setting options the old values of the changed options are (invisibly) returned as a list.

Note

If `...` is missing and `restore.defaults` = `FALSE`, all options and their current values will be returned as a list.

See Also

`print.BibEntry`, `BibEntry`, `options`

Examples

```
BibOptions()
BibOptions("first.inits", "bib.style")

oldopts <- BibOptions(first.inits = FALSE, bib.style = "authoryear")
oldopts
BibOptions(oldopts)

BibOptions(restore.defaults = TRUE)
```

---

c.BibEntry *Combine BibEntry objects.*

Description

Combines mutliple BibEntry objects into a single one.

Usage

```
## S3 method for class 'BibEntry'
c(..., recursive = FALSE)
```

Arguments

...	- BibEntry objects to be concatenated.
recursive	- logical; ignored.

Value

a single BibEntry object.

Note

c will remove all attributes besides class.

No checking for duplicate entries is performed though keys will be made unique.

See Also

Other operators: $.BibEntry(), $<-.BibEntry(), +.BibEntry(), [.BibEntry(), [<-.BibEntry(), [[.BibEntry(), [[<-.BibEntry()

Examples

```
bib <- c(BibEntry(bibtype = "article", key = "mclean2014a", title = "My New Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-01"),
  BibEntry(bibtype = "article", key = "mclean2014b",
  title = "My Newer Article", author = "Mathew W. McLean", journaltitle = "The Journal",
  date = "2014-02"))
```

Cite *Cite a BibEntry object in text and print all citations*

Description

The Cite functions allow for citing a BibEntry object in text. The PrintBibliography function allows for printing the bibliography of all the cited entries. The NoCite function adds references to the bibliography without including a citation. These functions are most useful when used in, e.g., a RMarkdown or RHTML document.

Usage

```
Cite(bib, ..., textual = FALSE, before = NULL, after = NULL, .opts = list())

PrintBibliography(bib, .opts = list(), start = 1, end = length(bib))

Citep(bib, ..., before = NULL, after = NULL, .opts = list())

AutoCite(bib, ..., before = NULL, after = NULL, .opts = list())

Citet(bib, ..., before = NULL, after = NULL, .opts = list())
```

```
TextCite(bib, ..., before = NULL, after = NULL, .opts = list())

NoCite(bib, ..., .opts = list())
```

Arguments

bib	a BibEntry or bibentry object
...	passed to SearchBib for indexing into bib. A character vector of keys, for example.
textual	logical; if TRUE, a "textual" citation is produced, i.e. what is produced by \citet in natbib and \textcite in BibLaTeX; otherwise, a parenthetical citation as \citep and \autocite.
before	string; optional text to display before the citation.
after	string; optional text to display after the citation.
.opts	list; See the relevant section in BibOptions for a description of all valid options for these functions.
start	Integer; specifying the index of the first citation to print. Useful for printing long bibliographies on multiple pages/slides.
end	Integer; specifying the index of the last citation to print. Useful for printing long bibliographies on multiple pages/slides.

Details

See the package vignettes and execute the examples below.

If bib.style = "alphabetic" or bib.style = "numeric", then sorting needs to be done at the start of the document prior to using a cite function as sorting is not done by the PrintBibliography function for those styles (specifying sorting in .opts is ignored in this case). If no sorting is done, the references are listed in the order they were cited in for those two styles.

If the ... argument to NoCite is identical to "*", then all references in bib are added to the bibliography without citations.

Value

For the cite functions: a character string containing the citation

PrintBibliography: The formatted list of references.

NoCite: no return value; invoked for its side-effect.

See Also

print.BibEntry, BibOptions, citeNatbib, the package vignettes bib <-

Examples

```
if (requireNamespace("bibtex")) {
    file <- system.file("Bib", "biblatexExamples.bib", package = "RefManageR")
    BibOptions(check.entries = FALSE)
    bib <- ReadBib(file)
    Citet(bib, 12)
    NoCite(bib, title = "Alkanethiolate")
    PrintBibliography(bib, .opts = list(style = "latex",
                      bib.style = "authoryear"))
}
## Not run:
  if (requireNamespace("bibtex")){
    Citep(bib, c("loh", "geer"), .opts = list(cite.style = "numeric"),
          before = "see e.g., ")
    Citet(bib, "loh", .opts = list(cite.style = "numeric", super = TRUE))
    AutoCite(bib, eprinttype = "arxiv", .opts = list(cite.style = "authoryear"))
    AutoCite(bib, eprinttype = "arxiv", .opts = list(cite.style = "pandoc"))
    Citep(bib, author = "kant")
    ## shorthand field in both entries gets used for numeric and alphabetic labels
    TextCite(bib, author = "kant", .opts = list(cite.style = "alphabetic"))
    TextCite(bib, author = "kant", .opts = list(cite.style = "numeric"))
    TextCite(bib, author = "kant", .opts = list(cite.style = "alphabetic",
             style = "html"))
    punct <- unlist(BibOptions("bibpunct"))
    punct[3:4] <- c("(", ")")
    TextCite(bib, 33, .opts = list(bibpunct = punct, cite.style = "alphabetic"))

    BibOptions(restore.defaults = TRUE)
  }

## End(Not run)
## Not run:
library(knitr)
## See also TestNumeric.Rmd and TestAlphabetic.Rmd for more examples
old.dir <- setwd(tdir <- tempdir())
doc <- system.file("Rmd", "TestRmd.Rmd", package = "RefManageR")
file.show(doc)
tmpfile <- tempfile(fileext = ".html", tmpdir = tdir)
knit2html(doc, tmpfile)
browseURL(tmpfile)

doc <- system.file("Rhtml", "TestAuthorYear.Rhtml", package = "RefManageR")
file.show(doc)
tmpfile <- tempfile(fileext = ".html", tmpdir = tdir)
knit2html(doc, tmpfile)
browseURL(tmpfile)
setwd(old.dir)
unlink(tdir)

## End(Not run)
```

GetBibEntryWithDOI *Lookup a Bibtex entry using a Digital Object Identifier*

Description

Uses the DOI System API to look up bibliography information given a set of DOIs.

Usage

```
GetBibEntryWithDOI(
  doi,
  temp.file = tempfile(fileext = ".bib"),
  delete.file = TRUE
)
```

Arguments

doi	character vector; DOIs to use to retrieve bibliographic information.
temp.file	string; a file to write the Bibtex data returned by the DOI System to.
delete.file	logical; should temp.file be deleted when the function exits?

Details

The bibliographic information returned by the search of the https://www.doi.org/ API is temporarily written to a file and then read back into R and return as a BibEntry object.

Value

an object of class BibEntry.

References

https://www.doi.org/tools.html

See Also

ReadCrossRef, BibEntry

Examples

```
if (interactive() && !httr::http_error("https://www.doi.org/"))
  GetBibEntryWithDOI(c("10.1016/j.iheduc.2003.11.004", "10.3998/3336451.0004.203"))
```

`GetPubMedByID` *Retrieve citation information from NCBI's Entrez for a set of PubMed IDs*

Description

Uses NCBI's E-Utilities to retrieve bibliographic information given a vector of PubMed ID's and returns the results as a BibEntry object.

Usage

```
GetPubMedByID(id, db = "pubmed", ...)
```

Arguments

`id`	character vector; PubMed ID's for searching NCBI's Entrez.
`db`	string; Entrez database to search.
`...`	additional parameters to use for the search. See the Entrez documentation listed in the *References*.

Value

a BibEntry object.

Note

Returned entries will have `bibtype` "Article" or "Book", unless a collection title is present – in which case the `bibtype` will be "InBook" – or there is no journal information returned for an article – in which case the `bibtype` will be "Misc".

References

https://www.ncbi.nlm.nih.gov/books/NBK25500/

See Also

Other pubmed: `GetPubMedRelated()`, `LookupPubMedID()`, `ReadCrossRef()`, `ReadPubMed()`

Examples

```
if (interactive() && !httr::http_error("https://eutils.ncbi.nlm.nih.gov/"))
  GetPubMedByID(c("11209037", "21245076"))
```

GetPubMedRelated *Retrieve related articles from PubMed using PubMed ID's*

Description

Searches PubMed for articles related to a set of PubMed ID's using NCBI's E-Utilities.

Usage

```
GetPubMedRelated(
  id,
  database = "pubmed",
  batch.mode = TRUE,
  max.results = 10,
  return.sim.scores = FALSE,
  return.related.ids = FALSE
)
```

Arguments

id	either a character vector of PubMed ID's or a BibEntry object, which is expected to have at least some entries with eprinttype = "pubmed" and eprint field specifying a PubMed ID.
database	string; the Entrez database to search
batch.mode	logical; if TRUE, the PubMed IDs in id are combined by Entrez when searching for linked IDs so that only one set of linked IDs is returned. If FALSE, a set of linked IDs is obtained for each ID in id. will be returned
max.results	numeric vector; the maximum number of results to return if batch.mode TRUE; or if batch.mode is FALSE, this should have the same length as id with each element giving the maximum number of results to return for the corresponding ID.
return.sim.scores	logical; Entrez returns a similarity score with each returned citation giving a measure of how similar the returned entry is to the ones specified by the query. If TRUE these scores are added to the returned BibEntry object in a field called 'score'.
return.related.ids	logical; should the original PubMed ID(s) that a returned entry is related to be stored in a field called 'PMIDrelated'.

Value

an object of class BibEntry.

References

https://www.ncbi.nlm.nih.gov/books/NBK25500/

See Also

Other pubmed: GetPubMedByID(), LookupPubMedID(), ReadCrossRef(), ReadPubMed()

Examples

```
if (interactive() && !httr::http_error("https://eutils.ncbi.nlm.nih.gov/")){
  file.name <- system.file("Bib", "RJC.bib", package="RefManageR")
  bib <- ReadBib(file.name)
  bib <- LookupPubMedID(bib[[101:102]])
  toBiblatex(GetPubMedRelated(bib, batch.mode = TRUE, max.results = 2,
  return.sim.scores = TRUE, return.related.ids = TRUE))
  GetPubMedRelated(bib, batch.mode = FALSE, max.results = c(2, 2))
}
```

---

head.BibEntry *Return the first or last part of a BibEntry object*

Description

Prints the first or last entries of a BibEntry object (via message) and returns them *invisibly* (via invisible).

Usage

```
## S3 method for class 'BibEntry'
head(x, n = 6L, suppress.messages = TRUE, ...)

## S3 method for class 'BibEntry'
tail(x, n = 6L, suppress.messages = TRUE, ...)
```

Arguments

x	an object of class BibEntry.
n	a single integer. If positive, size for the resulting object: number of elements for a vector (including lists), rows for a matrix or data frame or lines for a function. If negative, all but the n last/first number of elements of x.
suppress.messages	boolean; should the head/tail entries be printed via message?
...	arguments to be passed to or from other methods.

Details

If suppress.messages is FALSE, the head/tail entries are output to the console along with some additional formatting for the 'bibtype' and 'key', in addition to invisibly returning the entries.

Value

an object of class BibEntry.

Examples

```
if (requireNamespace("bibtex")) {
    file <- system.file("Bib", "biblatexExamples.bib", package = "RefManageR")
    BibOptions(check.entries = FALSE)
    bib <- ReadBib(file)
    tail(bib, 2, suppress.messages = FALSE)
    bib <- head(bib, 1, suppress.messages = TRUE)
}
```

---

levels.BibEntry *Extract all fields present in a BibEntry object*

Description

These functions return a list of all fields present in a BibEntry object.

Usage

```
## S3 method for class 'BibEntry'
levels(x)

fields(x)
```

Arguments

x	a BibEntry object.

Value

a list with the same length as x of character vectors giving the fields present in each entry of a BibEntry object.

Note

The only difference between `fields` and `levels` is that `levels` returns a list with element names corresponding to entry keys.

Examples

```
bib <- as.BibEntry(list(c(bibtype = "Article", key = "mclean2014a", title = "My New Article",
  author = "Mathew W. McLean",
  journaltitle = "The Journal", date = "2014-01"), c(bibtype = "Book", key = "mclean2014b",
  title = "My New Book", editor = "Mathew W. McLean", ISBN = "247123837", date = "2014-02")))
fields(bib)
levels(bib)
```

LookupPubMedID *Retrieve PubMed ID's for a BibEntry object*

Description

Uses the NCBI E-utilities to to search for PubMed ID's for citations stored in a BibEntry object.

Usage

```
LookupPubMedID(bib, index)
```

Arguments

bib	a bibentry object
index	indices specifying which entries of bib will be searched for. If missing, all entries are searched for.

Details

For each entry a citation string is created using the fields journaltitle/journal, date/year, volume, pages, and author; and these strings are then used to search the NCBI database for PubMed ID's.

If an ID is found for an entry, the entry is updated so that the eprinttype field is assigned the value "pubmed" and the eprint field is assigned the ID.

Value

a BibEntry object - bib with additional eprinttype and eprint fields when the search is successful for an entry.

See Also

Other pubmed: GetPubMedByID(), GetPubMedRelated(), ReadCrossRef(), ReadPubMed()

Examples

```
if (interactive() && !httr::http_error("https://eutils.ncbi.nlm.nih.gov/")){
  file.name <- system.file("Bib", "RJC.bib", package = "RefManageR")
  bib <- ReadBib(file.name)
  LookupPubMedID(bib[[101:102]])
}
```

`names<-.BibEntry` *Names (keys) of a BibEntry object*

Description

Functions to get and set the keys of an object of class BibEntry

Usage

```
## S3 replacement method for class 'BibEntry'
names(x) <- value

## S3 method for class 'BibEntry'
names(x)
```

Arguments

`x`	an object of class BibEntry
`value`	character vector of new key values to replace into `x`

Value

`names<-` the updated BibEntry object.

`names` - character vector of the keys of the BibEntry object.

Author(s)

McLean, M. W. <mathew.w.mclean@gmail.com>

Examples

```
if (requireNamespace("bibtex")) {
    bib <- ReadBib(system.file("Bib", "test.bib", package = "RefManageR"))
    names(bib)
    names(bib)[1] <- 'newkey'
}
```

`open.BibEntry` *Open BibEntry in PDF viewer or web browser.*

Description

Attempts to open a connection to an entry in a BibEntry object using fields such as 'file', 'DOI', 'eprint' + 'eprinttype', and 'URL'.

Usage

```
## S3 method for class 'BibEntry'
open(
  con,
  entry = 1L,
  open.field = c("file", "url", "eprint", "doi"),
  viewer,
  ...
)
```

Arguments

`con`	BibEntry object to extract connections from.
`entry`	numeric index or character key of entry in `bib` to open.
`open.field`	character vector of fields to use in `bib` to open the BibEntry. Possible fields are any combination of "file", "url", "eprint", or "doi". "eprint" is implemented for `eprinttype`= "JSTOR", "PubMed", or "arXiv". When multiple fields are specified, they are tried in the order they appear in the vector.
`viewer`	character string giving the name of the program to be used as hypertext browser. It should be in the PATH, or a full path specified. Alternatively, an R function to be called to invoke the browser. Defaults to `getOptions("pdfviewer")` if `open.field` = `"file"` and `getOptions("browser")`, otherwise.
`...`	not used.

Author(s)

McLean, M. W. <mathew.w.mclean@gmail.com>

See Also

`browseURL`

Examples

```
## Not run:
if (requireNamespace("bibtex")) {
    testbib <- ReadBib(system.file("REFERENCES.bib", package="bibtex"))
    open(testbib)
```

```
    testbib$file <- file.path(R.home("doc/manual"), "R-intro.pdf")
    open(testbib)
}

## End(Not run)
```

---

print.BibEntry *Print BibLaTeX bibliography Entries*

Description

Prints bibliographic information stored in BibEntry objects in BibLaTeX style

Usage

```
## S3 method for class 'BibEntry'
print(x, .opts = list(), ...)
```

Arguments

x	a BibEntry object
.opts	a list of formatting options from BibOptions. Possible options are • style - character string naming the printing style. Possible values are plain text (style "text"), BibTeX ("Bibtex"), BibLaTeX ("Biblatex"), a mixture of plain text and BibTeX as traditionally used for citations ("citation"), HTML ("html"), LaTeX ("latex"), "markdown", "yaml", R code ("R"), and a simple copy of the textVersion elements (style "textVersion", see BibEntry) • bib.style - character string specifying BibLaTeX style to use for formatting references. Possible values are "numeric" (default), "authoryear", "authortitle", "alphabetic", "draft". See section 3.3.2 of the BibLaTeX manual. • sorting - how should the entries in x be sorted? See sort.BibEntry. • max.names - maximum number of names to display for name list fields before truncation with "et al.". • first.inits - logical; if true only initials of given names are printed, otherwise full names are used. • dashed - logical; for .bibstyle = "authoryear" or .bibstyle = "authoryear" only, if TRUE duplicate author and editor lists are replaced with "—" when printed. • no.print.fields character vector; fields that should not be printed, e.g., doi, url, isbn, etc.
...	extra parameters to pass to the renderer.

Note

setting max.names to `value` is equivalent to setting `maxnames=value` and `minnames=value` in BibLaTeX.

Custom BibLaTeX styles may be defined using the function `bibstyle`. To fully support BibLaTeX, the created environment must have functions for formatting each of the entry types described in `BibEntry`.

References

Lehman, Philipp and Kime, Philip and Boruvka, Audrey and Wright, J. (2013). The biblatex Package. http://mirrors.ibiblio.org/CTAN/macros/latex/contrib/biblatex/doc/biblatex.pdf.

See Also

`BibEntry`, `ReadBib`, `sort.BibEntry`

Examples

```
if (requireNamespace("bibtex")) {
    file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
    bib <- suppressMessages(ReadBib(file.name))
    print(bib[author="aristotle"], .opts = list(bib.style = "numeric"))
    print(bib[55:57], .opts = list(bib.style = "authortitle", first.inits = FALSE))
    print(bib[80:88], .opts = list(bib.style = "alphabetic", max.names = 1,
          no.print.fields = "issn"))
    print(bib[32:36], .opts = list(bib.style = "draft"))
    oldopts <- BibOptions(bib.style = "authoryear", dashed = TRUE, sorting = "ydnt")
    bib[editor = "westfahl"]
    BibOptions(oldopts)
}
```

`ReadBib` *BibLaTeX/BibTeX .bib file parser*

Description

Parser for bibliography databases in the bib format containing either BibLaTeX or BibTeX entries.

Usage

```
ReadBib(
  file,
  .Encoding = "UTF-8",
  header = if (length(preamble)) paste(preamble, sep = "\n") else "",
  footer = "",
  check = BibOptions()$check.entries
)
```

Arguments

file	string; bib file to parse.
.Encoding	encoding
header	header of the citation list. By default this is made from the Preamble entries found in the bib file.
footer	footer of the citation list.
check	"error", "warn", or logical FALSE. What action should be taken if an entry is missing required fields? FALSE means no checking is done, "warn" means entry is added with an error. "error" means the entry will not be added. See BibOptions.

Note

Date fields are parsed using the locale specified by Sys.getlocale("LC_TIME"). To read a bib file with character 'month' fields in a language other than the current locale, Sys.setlocale should be used to change 'LC_TIME'' to match the bib file before calling ReadBib.

Keys will be made unique by calling make.unique with sep = ":".

Author(s)

McLean, M. W., based on code in bibtex package by Francois, R.

See Also

read.bib in package bibtex

Examples

```
bib.text <- "@Manual{mclean2014,
  author = {Mathew William McLean},
  title = {Straightforward Bibliography Management in R Using the RefManager Package},
  note = {arXiv: 1403.2036 [cs.DL]},
  year = {2014},
  url = {https://arxiv.org/abs/1403.2036},
}"
tfile <- tempfile(fileext = ".bib")
writeLines(bib.text, tfile)
ReadBib(tfile)
unlink(tfile)
## Not run:
    file.name <- system.file("Bib", "RJC.bib", package="RefManageR")
    bib <- ReadBib(file.name)

## End(Not run)
```

ReadCrossRef *Search CrossRef for citations.*

Description

Provides an interface to the CrossRef API, searching for citations given a string query. Results are written to a bib file, read back into R using WriteBib, and returned as a BibEntry object.

Usage

```
ReadCrossRef(
  query = "",
  filter = list(),
  limit = 5,
  offset = 0,
  sort = "relevance",
  year = NULL,
  min.relevance = 2,
  temp.file = tempfile(fileext = ".bib"),
  delete.file = TRUE,
  verbose = FALSE,
  use.old.api = FALSE
)
```

Arguments

query	string; search term
filter	named list of possible filters; see Details and References; ignored if use.old.api = TRUE
limit	numeric; maximum number of entries to return
offset	numeric; CrossRef will not return the first offset results (default 0); ignored if use.old.api = TRUE
sort	string; how specifying how the results from CrossRef should be sorted. Possible values when use.old.api = FALSE are "score" (default; same as "relevance"), "updated", "deposited", "indexed", or "published"; see the references
year	numeric; if specified, only results from this year will be returned.
min.relevance	numeric; only results with a CrossRef-assigned relevance score at least this high will be returned.
temp.file	string; file name to use for storing Bibtex information returned by CrossRef.
delete.file	boolean; should the bib file be deleted on exit?
verbose	boolean; if TRUE, additional messages are output regarding the results of the query.
use.old.api	boolean; should the older CrossRef API be used for the search? NO LONGER SUPPORTED, all queries need to use the new API.

Details

When use.old.api = TRUE, the query HTTP request only returns DOIs, which are then used to make HTTP requests for the corresponding BibTeX entries from CrossRef; when use.old.api = FALSE, the query HTTP request is parsed to create the BibEntry object (i.e. there are less HTTP requests when using the new API).

CrossRef assigns a score between 0 and 100 based on how relevant a reference seems to be to your query. The *old* API documentation warns that while false negatives are unlikely, the search can be prone to false positives. Hence, setting min.revelance to a high value may be necessary if use.old.api = TRUE. In some instances with the old API, no score is returned, if this happens, the entries are added with a message indicating that no score was available.

Possible values for the *names* in filter are "has-funder", "funder", "prefix", "member", "from-index-date", "until-index-date", "from-deposit-date", "until-deposit-date", "from-update-date", "until-update-date", "from-created-date", "until-created-date", "from-pub-date", "until-pub-date", "has-license", "license.url", "license.version", "license.delay", "has-full-text", "full-text.version", "full-text.type", "public-references", "has-references", "has-archive", "archive", "has-orcid", "orcid", "issn", "type", "directory", "doi", "updates", "is-update", "has-update-policy", "container-title", "publisher-name", "category-name", "type-name", "award.number", "award.funder", "assertion-group", "assertion", "affiliation", "has-affiliation", "alternative-id", and "article-number". See the first reference for a description of their meanings.

Value

An object of class BibEntry.

Note

The entries returned by Crossref are frequently missing fields required by BibTeX, if you want the entries to be returned anyway, set BibOptions()$check.entries to FALSE or "warn"

Fields "score" (the relevancy score) and "license" will be returned when use.old.api = FALSE.

References

Newer API: https://github.com/CrossRef/rest-api-doc/blob/master/rest_api.md, Older API: https://search.crossref.org/help/api

See Also

ReadZotero, BibEntry, package rcrossref for larger queries and deep paging

Other pubmed: GetPubMedByID(), GetPubMedRelated(), LookupPubMedID(), ReadPubMed()

Examples

```
if (interactive() && !httr::http_error("https://search.crossref.org/")){
  BibOptions(check.entries = FALSE)
  ## 3 results from the American Statistical Association involving "regression"
  ReadCrossRef("regression", filter = list(prefix="10.1198"), limit = 3)
```

```
  ## Some JRSS-B papers published in 2010 or later, note the quotes for filter
  ##   names with hypens
  ReadCrossRef(filter = list(issn = "1467-9868", "from-pub-date" = 2010),
               limit = 2, min.relevance = 0)

  ## Articles published by Institute of Mathematical Statistics
  ReadCrossRef(filter = list(prefix = "10.1214"), limit = 5, min.relevance = 0)

  ## old API
  ReadCrossRef(query = 'rj carroll measurement error', limit = 2, sort = "relevance",
    min.relevance = 80, use.old.api = TRUE)

  ReadCrossRef(query = 'carroll journal of the american statistical association',
    year = 2012, limit = 2, use.old.api = TRUE)
}
```

ReadGS *Import book and article references from a public Google Scholar profile by ID.*

Description

This function will create a BibEntry object for up to 100 references from a provided Google Scholar ID, if the profile is public. The number of citations for each entry will also be imported.

Usage

```
ReadGS(
  scholar.id,
  start = 0,
  limit = 100,
  sort.by.date = FALSE,
  .Encoding = "UTF-8",
  check.entries = BibOptions()$check.entries
)
```

Arguments

scholar.id	character; the Google Scholar ID from which citations will be imported. The ID can by found by visiting an author's Google Scholar profile and noting the value in the uri for the "user" parameter.
start	numeric; index of first citation to include.
limit	numeric; maximum number of results to return. Cannot exceed 100.
sort.by.date	boolean; if true, newest citations are imported first; otherwise, most cited works are imported first.
.Encoding	character; text encoding to use for importing the results and creating the bib entries.

check.entries What should be done with incomplete entries (those containing "..." due to long fields)? Either FALSE to add them anyway, "warn" to add with a warning, or any other value to drop the entry with a message and continue processing the remaining entries.

Details

This function creates BibTeX entries from an author's Google Scholar page. If the function finds numbers corresponding to volume/number/pages of a journal article, an 'Article' entry is created. If an arXiv identifier is found, a 'Misc' entry is created with eprint, eprinttype, and url fields. Otherwise, a 'TechReport' entry is created; unless the entry has more than ten citations, in which case a 'Book' entry is created.

Long author lists, long titles, and long journal/publisher names can all lead to these fields being incomplete for a particular entry. When this occurs, these entries are either dropped or added with a warning depending on the value of the check.entries argument.

Value

An object of class BibEntry. If the entry has any citations, the number of citations is stored in a field 'cites'.

Note

Read Google's Terms of Service before using.

It is not possible to automatically import BibTeX entries directly from Google Scholar as no API is available and this violates their Terms of Service.

See Also

BibEntry

Examples

```
if (interactive() && !httr::http_error("https://scholar.google.com")){
  ## R. J. Carroll's ten newest publications
  ReadGS(scholar.id = "CJOHNoQAAAAJ", limit = 10, sort.by.date = TRUE)

  ## Matthias Katzfu\ss
  BibOptions(check.entries = "warn")
  kat.bib <- ReadGS(scholar.id = "vqW0UqUAAAAJ")

  ## retrieve GS citation counts stored in field 'cites'
  kat.bib$cites
}
```

ReadPDFs *Create bibliographic information from PDF Metadata.*

Description

This function creates bibliographic information by reading the Metadata and text of PDFs stored in a user specified directory using Poppler (https://poppler.freedesktop.org/). IF requested, the function first searches for DOIs and downloads BibTeX entries from ReadCrossRef if DOIs are found. If this is not requested or a DOI is not found for an entry, an attempt is made to build a BibTeX entry from the metadata and text.

Usage

```
ReadPDFs(
  path,
  .enc = "UTF-8",
  recursive = TRUE,
  use.crossref = TRUE,
  use.metadata = TRUE,
  progress = FALSE
)
```

Arguments

path	character; path to directory containing pdfs or filename of one pdf. normalizePath is used on the specified path
.enc	character; text encoding to use for reading pdf and creating BibEntry object. Available encodings for Poppler can be found using system("pdfinfo -listenc"). The encoding must also be listed in iconvlist().
recursive	logical; same as list.files. Should pdfs in subdirectories of path be used?
use.crossref	logical; should an attempt be made to download bibliographic information from CrossRef if any Document Object Identifiers (DOIs) are found? This is only supported if the Suggeseted package bibtex is found.
use.metadata	logical; should the PDF metadata also be used to help create entries?
progress	logical; should progress bar be generated when fetching from CrossRef?

Details

This function requires that the pdfinfo utility from Poppler PDF https://poppler.freedesktop.org/ be installed.

This function will create only Article or Misc BibTeX entries.

The absolute path to each file will be stored in the bib entry in a field called 'file', which is recognized by BibLaTeX (though not printed by any standard style) and can be used by the open.BibEntry function to open the PDF in the default viewer.

If the keywords metadata field is available, it will be added to the bib entry in a field 'keywords', which is recognized by BibLaTeX.

Value

An object of class BibEntry.

References

https://poppler.freedesktop.org/

See Also

ReadCrossRef, BibEntry, open.BibEntry

Examples

```
## Not run:
path <- system.file("doc", package = "RefManageR")
ReadPDFs(path)

## End(Not run)
```

ReadPubMed *Search NCBI's E-Utilities for citation information*

Description

This function takes a query and searches an Entrez database for references using NCBI's E-Utilities, returning the results in a BibEntry object.

Usage

```
ReadPubMed(query, database = "PubMed", ...)
```

Arguments

query	string; search term.
database	string; the Entrez database to search.
...	additional parameters to use for the search. See the *Details*.

Details

Optional additional parameters to pass to the server include

- retstart - index of the first retrieved ID that should be included in the results.
- retmax - maximum number of IDs the server will return (default 20).
- field - limits the query to search only the specified field (e.g. "title").
- datetype - type of date to use when limiting search by dates. E.g. "mdat" for modification date or "pdat" for publication date.

- `reldate` - integer; only items that have (`datetype`) date values within `reldate` *days* of the current date will be returned.
- `mindate`, `maxdate` - date ranges to restrict search results. Possible formats are "YYYY", "YYYY/MM", and "YYYY/MM/DD".

Value

an object of class BibEntry.

Note

The returned entries will have type either 'Article' or 'Misc' depending on whether journal information was retrieved. See the Entrez documentation listed in the *References*.

The language of the entry will be returned in the field "language" and the abstract will be returned in the field "abstract", if they are available.

References

https://www.ncbi.nlm.nih.gov/books/NBK25499/#chapter4.ESearch

See Also

Other pubmed: `GetPubMedByID()`, `GetPubMedRelated()`, `LookupPubMedID()`, `ReadCrossRef()`

Examples

```
if (interactive() && !httr::http_error("https://eutils.ncbi.nlm.nih.gov/"))
  ReadPubMed(query = "raymond carroll measurement error", retmax = 5, mindate = 1990)
```

ReadZotero *Get Bibliography Information From a Zotero Library.*

Description

Get Bibliography Information From a Zotero Library.

Usage

```
ReadZotero(
  user,
  group,
  .params,
  temp.file = tempfile(fileext = ".bib"),
  delete.file = TRUE
)
```

Arguments

`user`	Zotero userID for use in calls to the Zotero API. This is not the same as your Zotero username. The userID for accessing user-owned libraries can be found at `https://www.zotero.org/settings/keys` after logging in.
`group`	Zotero groupID for use in calls to the Zotero API. Only one of `user` and `group` should be specified; `group` will be ignored if both are specified.
`.params`	A *named* list of parameters to use in requests to the Zotero API with possible values • q - Search string to use to search the library • qmode - Search mode. Default is "titleCreatorYear". Use "everything" to include full-text content in search. • key - API key. This must be specified to access non-public libraries. • collection - name of a specific collection within the library to search • itemType - type of entry to search for; e.g., "book" or "journalArticle" • tag - name of tag to search for in library • limit - maximum number of entries to return • start - index of first entry to return
`temp.file`	character; file name where the BibTeX data returned by Zotero will be temporarily written.
`delete.file`	boolean; should `temp.file` be removed on exit?

Value

An object of class BibEntry

References

`https://www.zotero.org/support/dev/server_api/v2/read_requests`

See Also

`BibEntry`

Examples

```
## Not run:
## first two entries in library with bayesian in title
ReadZotero(user = "1648676", .params = list(q = "bayesian",
  key = "7lhgvcwVq60CDi7E68FyE3br", limit=2))

## Search specific collection
## collection key can be found by reading uri when collection is selected in Zotero
ReadZotero(user = "1648676", .params=list(q = "yu", key = "7lhgvcwVq60CDi7E68FyE3br",
  collection = "3STEQRNU"))

## Search by tag
## Notice the issue with how Zotero uses a TechReport entry for arXiv manuscripts
## This is one instance where the added fields of BibLaTeX are useful
```

```
ReadZotero(user = "1648676", .params=list(key = "7lhgvcwVq60CDi7E68FyE3br",
  tag = "Statistics - Machine Learning"))

## To read these in you must set check.entries to FALSE or "warn"
old.opts <- BibOptions(check.entries = FALSE)
length(ReadZotero(user = "1648676", .params = list(key = "7lhgvcwVq60CDi7E68FyE3br",
  tag = "Statistics - Machine Learning")))

## Example using groups
ReadZotero(group = "13495", .params = list(q = "Schmidhuber",
  collection = "QU23T27Q"))
BibOptions(old.opts)

## End(Not run)
```

`RelistBibEntry` *Flatten and unflatten BibEntry objects*

Description

`RelistBibEntry` unflattens a BibEntry object that has been flattened with `unlist`.

`unlist` flattens a BibEntry object to a single list where every field (including `bibtype` and `key`) of every entry is a separate element in the list.

Usage

```
RelistBibEntry(flesh, skeleton = NULL)

## S3 method for class 'BibEntry'
unlist(x, recursive = FALSE, use.names = TRUE)
```

Arguments

`flesh`	list; an `unlisted` BibEntry object
`skeleton`	currently ignored
`x`	a BibEntry object to flatten
`recursive`	ignored.
`use.names`	ignored.

Details

`RelistBibEntry` is only intended for use with `unlisted` BibEntry objects.

Value

`RelistBibEntry` - an object of class BibEntry

For `unlist`, a list with bib entries collapsed into a single list.

Note

The names of the list elements from an unlisted BibEntry object will not be unique. To do this see `make.unique`.

See Also

`as.BibEntry`

Examples

```
bib <- list(c(bibtype = "article", key = "mclean2014a", title = "My New Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-01"),
  c(bibtype = "article", key = "mclean2014b", title = "My Newer Article",
  author = "Mathew W. McLean", journaltitle = "The Journal", date = "2014-02"))
bib <- as.BibEntry(bib)
unlist(bib)
RelistBibEntry(unlist(bib))
```

`sort.BibEntry` *Sort a BibEntry Object*

Description

Sorts a `BibEntry` object by specified fields. The possible fields used for sorting and the order they are used in correspond with the options available in BibLaTeX.

Usage

```
## S3 method for class 'BibEntry'
sort(
  x,
  decreasing = FALSE,
  sorting = BibOptions()$sorting,
  .bibstyle = BibOptions()$bib.style,
  ...
)
```

Arguments

`x`	an object of class BibEntry
`decreasing`	logical; should the sort be increasing or decreasing?
`sorting`	sort method to use, see **Details**.
`.bibstyle`	bibliography style; used when `sort` is called by `print.BibEntry`
`...`	internal use only

Details

The possible values for argument `sorting` are

- nty - sort by name, then by title, then by year
- nyt - sort by name, then by year, then title
- nyvt - sort by name, year, volume, title
- anyt - sort by alphabetic label, name, year, title
- anyvt - sort by alphabetic label, name, year, volume, title
- ynt - sort by year, name, title
- ydnt - sort by year (descending), name, title
- debug - sort by keys
- none - no sorting is performed

All sorting methods first consider the field presort, if available. Entries with no presort field are assigned presort value "mm". Next the sortkey field is used.

When sorting by name, the sortname field is used first. If it is not present, the author field is used, if that is not present editor is used, and if that is not present translator is used. All of these fields are affected by the value of `max.names` in .BibOptions()$max.names.

When sorting by title, first the field sorttitle is considered. Similarly, when sorting by year, the field sortyear is first considered.

When sorting by volume, if the field is present it is padded to four digits with leading zeros; otherwise, the string "0000" is used.

When sorting by alphabetic label, the labels that would be generating with the "alphabetic" bibstyle are used. First the shorthand field is considered, then label, then shortauthor, shorteditor, author, editor, and translator. Refer to the BibLaTeX manual Sections 3.1.2.1 and 3.5 and Appendix C.2 for more information.

Value

the sorted BibEntry object

References

Lehman, Philipp and Kime, Philip and Boruvka, Audrey and Wright, J. (2013). The biblatex Package. http://mirrors.ibiblio.org/CTAN/macros/latex/contrib/biblatex/doc/biblatex.pdf.

See Also

`BibEntry`, `print.BibEntry`, `order`

Examples

```
if (requireNamespace("bibtex")) {
    file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
    bib <- suppressMessages(ReadBib(file.name)[[70:73]])
    BibOptions(sorting = "none")
    bib
    sort(bib, sorting = "nyt")
    sort(bib, sorting = "ynt")
    BibOptions(restore.defaults = TRUE)
}
```

`toBiblatex` *Convert BibEntry objects to BibTeX or BibLaTeX*

Description

toBiblatex converts a BibEntry object to character vectors with BibLaTeX markup. toBibtex will convert a BibEntry object to character vectors with BibTeX markup, converting some BibLaTeX fields and all entry types that are not supported by BibTeX to ones that are supported.

Usage

```
toBiblatex(object, ...)

## S3 method for class 'BibEntry'
toBibtex(
  object,
  note.replace.field = c("urldate", "pubsate", "addendum"),
  extra.fields = NULL,
  ...
)
```

Arguments

`object`	an object of class BibEntry to be converted
`...`	ignored
`note.replace.field`	a character vector of BibLaTeX fields. When converting an entry to BibTeX, the first field in the entry that matches one specified in this vector will be added to the note field, *if* the note field is not already present
`extra.fields`	character vector; fields that are not supported in standard BibTeX styles are by default dropped in the result return by the toBibtex function. Any fields specified in extra.fields will *not* be dropped if present in an entry.

Details

toBiblatex converts the BibEntry object to a vector containing the corresponding BibLaTeX file, it ensures the name list fields (e.g. author and editor) are formatted properly to be read by bibtex and biber and otherwise prints all fields as is, thus it is similar to `toBibtex`.

toBibtex will attempt to convert BibLaTeX entries to a format that can be read by bibtex. Any fields not supported by bibtex are dropped unless they are specified in `extra.fields`. The fields below, if they are present, are converted as described and added to a bibtex supported field, unless that field is already present.

- date - The `date` field, if present will be truncated to a year and added to the `year` field, if it is not already present. If a month is specified with the date, it will be added to the `month` field.
- journaltitle - Will be changed to journal, if it is not already present
- location - Will be changed to address
- institution - Converted to `school` for thesis entries
- sortkey - Converted to `key`
- maintitle - Converted to `series`
- issuetitle - Converted to `booktitle`
- eventtitle - Converted to `booktitle`
- eprinttype - Converted to `archiveprefix` (for arXiv references)
- eprintclass - Converted to `primaryclass` (for arXiv references)

If no `note` field is present, the note.replace.field can be used to specified BibLaTeX fields that can be looked for and added to the note field if they are present.

BibLaTeX entry types that are not supported by bibtex are converted by toBibtex as follows "mvbook" = "Book", "bookinbook" = "InBook", "suppbook" = "InBook",

- MvBook,Collection,MvCollection,Reference,MvReference,Proceedings,MvProceedings,Periodical - to Book
- BookInBook,SuppBook,InReference,SuppPeriodical - to InBook
- report,patent - to TechReport
- SuppCollection - to InCollection
- thesis - to MastersThesis if `type` = `mathesis`, else to PhdThesis
- *rest* - to Misc

Value

an object of class "Bibtex" - character vectors where each element holds one line of a BibTeX or BibLaTeX file

Author(s)

McLean, M. W. <mathew.w.mclean@gmail.com>

See Also

`toBibtex`, `BibEntry`, `print.BibEntry`

Examples

```
if (requireNamespace("bibtex")) {
    file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
    bib <- suppressMessages(ReadBib(file.name))
    toBiblatex(bib[70:72])
    toBibtex(bib[70:72])
}
```

`UpdateFieldName` *Rename a field in a BibEntry object.*

Description

This function will rename a field, in every entry where it is present, in a BibEntry object.

Usage

```
UpdateFieldName(x, old.field, new.field)
```

Arguments

`x`	- a BibEntry object
`old.field`	- string; the current name of the field to be renamed
`new.field`	- string; the new name to replace `old.field`

Value

`x`, with the renamed field.

Examples

```
bib <- as.BibEntry(list(c(bibtype = "article", key = "mclean2014a", title = "My New Article",
  author = "Mathew W. McLean", journal = "The Journal", date = "2014-01"),
  c(bibtype = "article", key = "mclean2014b", title = "My Newer Article",
    author = "Mathew W. McLean", journal = "The Journal", date = "2014-02")))
bib <- UpdateFieldName(bib, "journal", "journaltitle")
toBiblatex(bib)
```

WriteBib *Create a BibTeX File from a BibEntry Object e Creates a Bibtex File from a BibEntry object for use with either BibTeX or BibLaTex.*

Description

Create a BibTeX File from a BibEntry Object e Creates a Bibtex File from a BibEntry object for use with either BibTeX or BibLaTex.

Usage

```
WriteBib(
  bib,
  file = "references.bib",
  biblatex = TRUE,
  append = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

bib	a BibEntry object to be written to file
file	character string naming a file, should; end in ".bib". Can be NULL, in which case the BibEntry object will be written to stdout.
biblatex	boolean; if TRUE, toBiblatex is used and no conversions of the BibEntry object are done; if FALSE entries will be converted as described in toBibtex.BibEntry.
append	as in write.bib in package bibtex
verbose	as in write.bib in package bibtex
...	additional arguments passed to writeLines

Value

bib - invisibly

Note

To write the contents of bib "as is", the argument biblatex should be TRUE, otherwise conversion is done as in toBibtex.BibEntry.

Author(s)

McLean, M. W. based on write.bib by Gaujoux, R. in package bibtex.

See Also

write.bib in package bibtex, ReadBib, toBibtex.BibEntry, toBiblatex, BibEntry

Examples

```
if (requireNamespace("bibtex")){
    bib <- BibEntry("Article", key = "Carroll_2012",
                    doi = "10.1080/01621459.2012.699793",
                    year = "2012", month = "sep",
                    volume = 107, number = 499,
                    pages = {1166--1177},
      author = "R. Carroll and A. Delaigle and P. Hall",
      title = "Deconvolution When Classifying Noisy Data ...",
      journal = "Journal of the American Statistical Association")

  ## Write bib if no server error and bibtex available
  if (length(bib)){
    tfile <- tempfile(fileext = ".bib")
    WriteBib(bib, tfile, biblatex = TRUE)
    identical(ReadBib(tfile), bib)
    unlink(tfile)
  }
}
```

`[.BibEntry` *Search BibEntry objects by field*

Description

Allows for searching and indexing a BibEntry object by fields, including names and dates. The extraction operator and the `SearchBib` function simply provide different interfaces to the same search functionality.

Usage

```
## S3 method for class 'BibEntry'
x[i, j, ..., drop = FALSE]

SearchBib(x, .opts = list(), ...)
```

Arguments

`x`	an object of class BibEntry
`i`	A named list or character vector of search terms with names corresponding to the field to search for the search term. Alternatively, a vector of entry key values or numeric or logical indices specifying which entries to extract.
`j`	A named list or character vector, as `i`. Entries matching the search specified by i *OR* matching the query specified by `j` will be return
`...`	arguments in the form `bib.field = search.term`, or as `j` list*s* or character vector*s* for additional searches. For `SearchBib`, can alternatively have same form as `i`.

drop — logical, should attributes besides class be dropped from result?

.opts — list of search options with name = value entries. Any option described in BibOptions is valid, with the following being the most relevant ones

- use.regex - logical; are the search terms regular expressions or should exact matching be used?
- ignore.case - logical; should case be ignored when comparing strings?
- match.date - how should the date fields date, urldate, eventdate, and origdate. Default is "year.only", so that months and days in dates are ignored when comparing. Currently, specifying any other value results the full date being used. See the Note section.
- match.author - character string; how should name fields be searched? If "family.only", only family names are compared; if "family.with.initials", family name and given name initials are used; if "exact", full names are used.
- return.ind - logical; if TRUE the returned object is numeric indices of match locations; otherwise, a BibEntry object is returned

Value

an object of class BibEntry (the results of the search/indexing), *or* if BibOptions()$return.ind=TRUE, the indices in x that match the search terms.

Note

The arguments to the SearchBib function that control certain search features can also be changed for the extraction operator by changing the corresponding option in the .BibOptions object; see BibOptions.

See Also

Other operators: $.BibEntry(), $<-.BibEntry(), +.BibEntry(), [<-.BibEntry(), [[.BibEntry(), [[<-.BibEntry(), c.BibEntry()

Examples

```
file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
bib <- suppressMessages(ReadBib(file.name))

## author search, default is to use family names only for matching
bib[author = "aristotle"]

## Aristotle references before 1925
bib[author="aristotle", date = "/1925"]

## Aristotle references before 1925 *OR* references with editor Westfahl
bib[list(author="aristotle", date = "/1925"),list(editor = "westfahl")]

## Change some searching and printing options and search for author
old.opts <- BibOptions(bib.style = "authoryear", match.author = "exact",
```

```
                        max.names = 99, first.inits = FALSE)
bib[author="Mart\u00edn, Jacinto and S\u00e1nchez, Alberto"]
BibOptions(old.opts)  ## reset options

## Not run:
  ## Some works of Raymond J. Carroll's
  file.name <- system.file("Bib", "RJC.bib", package="RefManageR")
  bib <- ReadBib(file.name)
  length(bib)

  ## index by key
  bib[c("chen2013using", "carroll1978distributions")]

  ## Papers with someone with family name Wang
  length(SearchBib(bib, author='Wang', .opts = list(match.author = "family")))

  ## Papers with Wang, N.
 length(SearchBib(bib, author='Wang, N.', .opts = list(match.author = "family.with.initials")))

  ## tech reports with Ruppert
  length(bib[author='ruppert',bibtype="report"])

  ##Carroll and Ruppert tech reports at UNC
  length(bib[author='ruppert',bibtype="report",institution="north carolina"])

  ## Carroll and Ruppert papers since leaving UNC
  length(SearchBib(bib, author='ruppert', date="1987-07/",
                   .opts = list(match.date = "exact")))

  ## Carroll and Ruppert papers NOT in the 1990's
 length(SearchBib(bib, author='ruppert', date = "!1990/1999"))
 identical(SearchBib(bib, author='ruppert', date = "!1990/1999"),
          SearchBib(bib, author='ruppert', year = "!1990/1999"))
 table(unlist(SearchBib(bib, author='ruppert', date="!1990/1999")$year))

 ## Carroll + Ruppert + Simpson
 length(bib[author="Carroll, R. J. and Simpson, D. G. and Ruppert, D."])

 ## Carroll + Ruppert OR Carroll + Simpson
 length(bib[author=c("Carroll, R. J. and Ruppert, D.", "Carroll, R. J. and Simpson, D. G.")])

 ## Carroll + Ruppert tech reports at UNC "OR" Carroll and Ruppert JASA papers
 length(bib[list(author='ruppert',bibtype="report",institution="north carolina"),
        list(author="ruppert",journal="journal of the american statistical association")])

## End(Not run)
```

[<-.BibEntry *Update Different Fields of Multiple Entries of a BibEntry Object*

Description

Assign new values for specified fields in a BibEntry object using a named character vector or list of named character vectors.

Usage

```
## S3 replacement method for class 'BibEntry'
x[i, j, ...] <- value
```

Arguments

x	- a BibEntry object.
i	- see [.BibEntry
j	- see [.BibEntry
...	- see [.BibEntry
value	- values to be assigned to x. To update one entry only, should be a named character vector with names corresponding to fields. To update multiple entries, should be a list of named character vectors. Can also be an object of class BibEntry.

Details

Date and name list fields should be in the format expected by Biblatex (see BibEntry).

To clear a field 'field_name' from an entry use field_name = "".

Value

an object of class BibEntry.

See Also

Other operators: $.BibEntry(), $<-.BibEntry(), +.BibEntry(), [.BibEntry(), [[.BibEntry(), [[<-.BibEntry(), c.BibEntry()

Examples

```
bib.text <- "@Manual{mclean2014,
  author = {Mathew William McLean},
  title = {Straightforward Bibliography Management in R Using the RefManager Package},
  note = {arXiv: 1403.2036 [cs.DL]},
  year = {2014},
  url = {https://arxiv.org/abs/1403.2036},
}"
tfile <- tempfile(fileext = ".bib")
writeLines(bib.text, tfile)
bib <- ReadBib(tfile)
bib[1] <- list(c(date = "2014-03", key = "mwm2014"))
bib
unlink(tfile)
```

```
## Not run:
    file.name <- system.file("Bib", "RJC.bib", package="RefManageR")
    bib <- ReadBib(file.name)
    print(bib[seq_len(3L)], .opts = list(sorting = "none", bib.style = "alphabetic"))
    ## add month to Serban et al., add URL and urldate to Jennings et al., and
    ##   add DOI and correct journal to Garcia et al.
    bib[seq_len(3L)] <- list(c(date="2013-12"),
                             c(url="https://bsb.eurasipjournals.com/content/2013/1/13",
                               urldate = "2014-02-02"),
                             c(doi="10.1093/bioinformatics/btt608",
                               journal = "Bioinformatics"))
    print(bib[seq_len(3L)], .opts = list(sorting = "none", bib.style = "alphabetic"))
    bib2 <- bib[seq_len(3L)]
    bib2[2:3] <- bib[5:6]
    bib2
    bib2[3] <- c(journal='', eprinttype = "arxiv", eprint = "1308.5427",
      eprintclass = "math.ST", pubstate = "submitted", bibtype = "misc")
    bib2

## End(Not run)
```

`[[.BibEntry` *Extract entries from a BibEntry object by index*

Description

Operator for extracting BibEntry objects by index.

Usage

```
## S3 method for class 'BibEntry'
x[[i, drop = FALSE]]
```

Arguments

`x`	a BibEntry object
`i`	numeric indices of entries to extract, or a character vector of keys corresponding to the entries to be extracted.
`drop`	logical, should attributes besides class be dropped from result?

Value

an object of class BibEntry.

Note

This method is different than the usual operator [[for lists in that a vector of indices may be specified.

This method behaves differently than the [operator for BibEntry objects in that it does not expand crossreferences when returning, so that a parent entry or xdata entry will be dropped if it is not also indexed when indexing the child entry.

This method is not affected by the value of `BibOptions()$return.ind`.

See Also

Other operators: `$.BibEntry()`, `$<-.BibEntry()`, `+.BibEntry()`, `[.BibEntry()`, `[<-.BibEntry()`, `[[<-.BibEntry()`, `c.BibEntry()`

Examples

```
if (requireNamespace("bibtex")) {
    file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
    bib <- suppressMessages(ReadBib(file.name))
    bib[[20:21]]
    bib[c("hyman", "loh")]

   ## Note this is FALSE because [[ does not inherit from the dropped parent entry while [ does.
    identical(bib[1], bib[[1]])
}
```

[[<-.BibEntry *Assign a BibEntry entry to another BibEntry object*

Description

Replace one entry in a BibEntry object with another

Usage

```
## S3 replacement method for class 'BibEntry'
x[[i]] <- value
```

Arguments

x	- a BibEntry object
i	- a numeric index or a string entry key
value	- a single entry BibEntry object or an object that can be coerced to BibEntry using `as.BibEntry`

Details

This function will replace the specified entry in x with the entry given by value. To replace multiple entries see `[<-.BibEntry`.

Value

an object of class BibEntry

See Also

Other operators: `$.BibEntry()`, `$<-.BibEntry()`, `+.BibEntry()`, `[.BibEntry()`, `[<-.BibEntry()`, `[[.BibEntry()`, `c.BibEntry()`

`$.BibEntry` *Extract fields from a BibEntry object*

Description

used to extract a single field from each entry in a BibEntry object

Usage

```
## S3 method for class 'BibEntry'
x$name
```

Arguments

`x`	an object of class BibEntry
`name`	the field to extract

Value

a named list of values for the field specified by name for each entry; `NULL` if the field is not present for a particular entry. The names attribute of the returned list contains the entry keys (potentially back-quoted).

Note

`name` may be "bibtype" to extract entry types or "key" to extract keys.

See Also

Other operators: `$<-.BibEntry()`, `+.BibEntry()`, `[.BibEntry()`, `[<-.BibEntry()`, `[[.BibEntry()`, `[[<-.BibEntry()`, `c.BibEntry()`

Examples

```
if (requireNamespace("bibtex")) {
    file.name <- system.file("Bib", "biblatexExamples.bib", package="RefManageR")
    bib <- suppressMessages(ReadBib(file.name))
    bib[[50:55]]$author
    bib[[seq_len(5)]]$bibtype
 }
```

`$<-.BibEntry` *Replace values for a particular field in a BibEntry object*

Description

Used to replace the values stored for a specified field in a BibEntry object.

Usage

```
## S3 replacement method for class 'BibEntry'
x$name <- value
```

Arguments

`x`	a BibEntry object
`name`	string; the field to assign the new values to.
`value`	character vector; the replacement field values to be assigned.

Value

an object of class BibEntry with the updated fields.

Note

The method expects date and name list fields to be in the format expected by Biblatex. The field specified by `name` does not have to be one currently in `x`.

See Also

Other operators: `$.BibEntry()`, `+.BibEntry()`, `[.BibEntry()`, `[<-.BibEntry()`, `[[.BibEntry()`, `[[<-.BibEntry()`, `c.BibEntry()`

Examples

```
bib <- BibEntry(bibtype = "misc", key = "mclean", author = "Mathew W. McLean",
  title = "My Work", year = "2012")
bib$year <- 2014
bib$author <- "McLean, M. W. and Carroll, R. J."
bib$url <- "https://example.com"
bib

bib <- c(bib, as.BibEntry(citation()))
bib[1]$author[2] <- person(c("Raymond", "J."), "Carroll")
bib$author
```

Index