Package 'handlr'

August 19, 2019

Title Convert Among Citation Formats

Description Converts among many citation formats, including 'BibTeX', 'Citeproc', 'Codemeta', 'RDF XML', 'RIS', and 'Schema.org'. A low level 'R6' class is provided, as well as stand-alone functions for each citation format for both read and write.

Version 0.2.0

License MIT + file LICENSE

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

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

Encoding UTF-8

Language en-US

Imports jsonlite, crul, xml2, RefManageR, urltools, mime

Suggests testthat, jsonld, data.table

X-schema.org-applicationCategory Metadata

X-schema.org-keywords doi, metadata, citation, bibtex, Crossref, Crosscite, Codemeta, RIS, Citeproc, RDF, XML, JSON

X-schema.org-isPartOf https://ropensci.org

RoxygenNote 6.1.1

NeedsCompilation no

Author Scott Chamberlain [aut, cre] (<https://orcid.org/0000-0003-1444-9135>)

Maintainer Scott Chamberlain <sckott@protonmail.com>

Repository CRAN

Date/Publication 2019-08-19 19:00:02 UTC

R topics documented:

`handlr-package` **Citation format converter**

Description

A tool for converting among citation formats

supported readers

- citeproc
- ris
- bibtex
- codemeta

supported writers

- citeproc
- ris
- bibtex
- schema.org
- rdfxml (requires suggested package `jsonld`)
- codemeta

links for citation formats

- citeproc: https://en.wikipedia.org/wiki/CiteProc
- codemeta: https://codemeta.github.io/
- ris: https://en.wikipedia.org/wiki/RIS_(file_format)
- bibtex: http://www.bibtex.org/
- schema.org: https://schema.org/
- rdfxml: https://en.wikipedia.org/wiki/RDF/XML

Author(s)

Scott Chamberlain <sckott@protonmail.com>

`bibtex_reader` *bibtex reader*

Description

bibtex reader

Usage

```
bibtex_reader(x)
```

Arguments

`x`	(character) a file path or a bibtex string

Value

an object of class `handl`; see handl for more

See Also

Other readers: `citeproc_reader`, `codemeta_reader`, `ris_reader`

Other bibtex: `bibtex_writer`

Examples

```
(z <- system.file('extdata/crossref.bib', package = "handlr"))
bibtex_reader(x = z)
(z <- system.file('extdata/bibtex.bib', package = "handlr"))
bibtex_reader(x = z)

# many at once
(z <- system.file('extdata/bib-many.bib', package = "handlr"))
bibtex_reader(x = z)
```

bibtex_writer *bibtex writer*

Description

bibtex writer

Usage

```
bibtex_writer(z, key = NULL)
```

Arguments

z	an object of class handl; see handl for more
key	(character) optional bibtex key to use. if NULL we attempt try the following fields in order: key, identifier, id, doi. if you pass in ouput from bibtex_reader() you're likely to have a key field, but otherwise probably not

Value

an object of class BibEntry

See Also

Other writers: citeproc_writer, codemeta_writer, rdf_xml_writer, ris_writer, schema_org_writer

Other bibtex: bibtex_reader

Examples

```
(z <- system.file('extdata/citeproc.json', package = "handlr"))
(tmp <- citeproc_reader(z))
bibtex_writer(z = tmp)
cat(bibtex_writer(z = tmp), sep = "\n")

(z <- system.file('extdata/bibtex2.bib', package = "handlr"))
z <- bibtex_reader(z)
bibtex_writer(z)
cat(bibtex_writer(z), sep = "\n")

# give a bibtex key
cat(bibtex_writer(z, "foobar89"), sep = "\n")

# many at once
(z <- system.file('extdata/bib-many.bib', package = "handlr"))
out <- bibtex_reader(x = z)
bibtex_writer(out)
```

c.handl *combine many handl objects*

Description

combine many handl objects

Usage

```
## S3 method for class 'handl'
c(...)
```

Arguments

...	one or more objects of class handl; see handl for more. all inputs must be of class handl. if the first input is not of class handl, you will not get back an object of class handl

Value

an object of class handl of length equal to number of handl objects passed in

Examples

```
z <- system.file('extdata/crossref.ris', package = "handlr")
cr <- ris_reader(z)
z <- system.file('extdata/peerj.ris', package = "handlr")
prj <- ris_reader(z)
res <- c(cr, prj)
res
invisible(lapply(bibtex_writer(res), cat, sep = "\n\n"))
```

citeproc_reader *citeproc reader*

Description

citeproc reader

Usage

```
citeproc_reader(x)
```

Arguments

x	(character) a file path or string

Value

an object of class handl; see handl for more

See Also

Other readers: bibtex_reader, codemeta_reader, ris_reader

Other citeproc: citeproc_writer

Examples

```
# single
z <- system.file('extdata/citeproc.json', package = "handlr")
citeproc_reader(x = z)

# many
z <- system.file('extdata/citeproc-many.json', package = "handlr")
citeproc_reader(x = z)
```

---

citeproc_writer *citeproc writer*

Description

citeproc writer

Usage

```
citeproc_writer(z, auto_unbox = TRUE, pretty = TRUE, ...)
```

Arguments

z	an object of class handl; see handl for more
auto_unbox	(logical) automatically "unbox" all atomic vectors of length 1 (default: TRUE). passed to jsonlite::toJSON()
pretty	(logical) adds indentation whitespace to JSON output (default: TRUE), passed to jsonlite::toJSON()
...	further params passed to jsonlite::toJSON()

Value

citeproc as JSON

See Also

Other writers: bibtex_writer, codemeta_writer, rdf_xml_writer, ris_writer, schema_org_writer

Other citeproc: citeproc_reader

Examples

```
z <- system.file('extdata/citeproc.json', package = "handlr")
(tmp <- citeproc_reader(z))
citeproc_writer(z = tmp)
citeproc_writer(z = tmp, pretty = FALSE)
cat(ris_writer(z = tmp))

# many
z <- system.file('extdata/citeproc-many.json', package = "handlr")
w <- citeproc_reader(x = z)
citeproc_writer(w)
```

---

codemeta_reader *codemeta reader*

Description

codemeta reader

Usage

```
codemeta_reader(x)
```

Arguments

x	(character) a file path or string (character or json)

Value

an object of class handl; see handl for more

See Also

Other readers: bibtex_reader, citeproc_reader, ris_reader

Other codemeta: codemeta_writer

Examples

```
# single
(z <- system.file('extdata/codemeta.json', package = "handlr"))
codemeta_reader(x = z)

# many
(z <- system.file('extdata/codemeta-many.json', package = "handlr"))
codemeta_reader(x = z)
```

codemeta_writer *codemeta writer*

Description

codemeta writer

Usage

```
codemeta_writer(z, auto_unbox = TRUE, pretty = TRUE, ...)
```

Arguments

z	an object of class handl; see handl for more
auto_unbox	(logical) automatically "unbox" all atomic vectors of length 1 (default: TRUE). passed to jsonlite::toJSON()
pretty	(logical) adds indentation whitespace to JSON output (default: TRUE), passed to jsonlite::toJSON()
...	further params passed to jsonlite::toJSON()

Value

an object of class json

See Also

Other writers: bibtex_writer, citeproc_writer, rdf_xml_writer, ris_writer, schema_org_writer

Other codemeta: codemeta_reader

Examples

```
(x <- system.file('extdata/crossref.bib', package = "handlr"))
(z <- bibtex_reader(x))
codemeta_writer(z)

# many citeproc to schema
z <- system.file('extdata/citeproc-many.json', package = "handlr")
w <- citeproc_reader(x = z)
codemeta_writer(w)
codemeta_writer(w, pretty = FALSE)
```

`handl` *handl object*

Description

handl object

Details

A `handl` object is what's returned from the reader functions, and what is passed to the writer functions. The `handl` object is a list, but using the `print.handl` method makes it look something like:

```
<handl>
  from: codemeta
  many: TRUE
  count: 2
  first 10
    id/doi: https://doi.org/10.5063%2ff1m61h5x
    id/doi: https://doi.org/10.5063%2ff1m61h5x
```

You can always `unclass()` the object to get the list itself.

The `handl` object follows https://github.com/datacite/bolognese, which uses the Crosscite format as its internal representation. Note that we don't currently supporting writing to or reading from Crosscite.

Details on each entry are stored in the named attributes:

- from: the data type the citations come from
- many: is there more than 1 citation?
- count: number of citations
- finally, some details of the first 10 are printed

If you have a `handl` object with 1 citation, it is a named list that you can access with normal key indexing. If the result is length > 1, the data is an unnamed list of named lists; the top level list is unnamed, with each list within it being named.

Each named list should have the following components:

- key: (string) a key for the citation, e.g., in a bibtex file
- id: (string) an id for the work being referenced, often a DOI
- type: (string) type of work
- bibtex_type: (string) bibtex type
- citeproc_type: (string) citeproc type
- ris_type: (string) ris type
- resource_type_general
- additional_type: (string) additional type

- doi: (string) DOI
- b_url: (string) additional URL
- title: (string) the title of the work
- author: (list) authors, with each author a named list of
  - type: type, typically "Person"
  - name: full name
  - givenName: given (first) name
  - familyName: family (last) name
- publisher: (string) the publisher name
- is_part_of: (list) what the work is published in, or part of, a named list with:
  - type: (string) the type of work
  - title: (string) title of the work, often a journal or edited book
  - issn: (string) the ISSN
- date_published: (string)
- volume: (string) the volume, if applicable
- first_page: (string) the first page
- last_page: (string) the last page
- description: (string) description of the work, often an abstract
- license: (string) license of the work, a named list
- state: (string) the state of the list

`HandlrClient` *handlr client*

Description

handlr client

Arguments

`x`	(character) a file path (the file must exist), a string containing contents of the citation, a DOI, or a DOI as a URL. See Details.
`...`	curl options passed on to crul::HttpClient

Details

Methods

- `read(format = NULL, ...)` - read input
  - `format`: one of citeproc, ris, bibtex, codemeta, or `NULL`. If `NULL`, we attempt to guess the format, and error if we can not guess
  - `...`: further args to the writer fxn, if any

- `write(format,file = NULL,...)` - write to std out or file
  - `format`: one of citeproc, ris, bibtex, rdfxml
  - `file`: a file path, if NULL to stdout. for `format=ris`, number of files must equal number of ris citations
  - `...`: further args to the writer fxn, if any
  - Note: If `$parsed` is `NULL` then it's likely `$read()` has not been run - in which case we attempt to run `$read()` to populate `$parsed`

The various inputs to the `x` parameter are handled in different ways:

- file: contents read from file, we grab file extension, and we guess format based on combination of contents and file extension because file extensions may belie what's in the file
- string: string read in, and we guess format based on contents of the string
- DOI: we request citeproc-json format from the Crossref API
- DOI url: we request citeproc-json format from the Crossref API

Examples

```
# read() can be run with format specified or not
# if format not given, we attempt to guess the format and then read
z <- system.file('extdata/citeproc.json', package = "handlr")
(x <- HandlrClient$new(x = z))
x$read()
x$read("citeproc")
x$parsed

# you can run read() then write()
# or just run write(), and read() will be run for you if possible
z <- system.file('extdata/citeproc.json', package = "handlr")
(x <- HandlrClient$new(x = z))
cat(x$write("ris"))

# read from a DOI as a url
if (interactive()) {
  (x <- HandlrClient$new('https://doi.org/10.7554/elife.01567'))
  x$parsed
  x$read()
  x$write('bibtex')
}

# read from a DOI
if (interactive()) {
  (x <- HandlrClient$new('10.7554/elife.01567'))
  x$parsed
  x$read()
  x$write('bibtex')
}

# read in citeproc, write out bibtex
z <- system.file('extdata/citeproc.json', package = "handlr")
(x <- HandlrClient$new(x = z))
```

```
x$path
x$ext
x$read("citeproc")
x$parsed
x$write("bibtex")
f <- tempfile(fileext = ".bib")
x$write("bibtex", file = f)
readLines(f)
unlink(f)

# read in ris, write out ris
z <- system.file('extdata/peerj.ris', package = "handlr")
(x <- HandlrClient$new(x = z))
x$path
x$format
x$read("ris")
x$parsed
x$write("ris")
cat(x$write("ris"))

# read in bibtex, write out ris
(z <- system.file('extdata/bibtex.bib', package = "handlr"))
(x <- HandlrClient$new(x = z))
x$path
x$format
x$read("bibtex")
x$parsed
x$write("ris")
cat(x$write("ris"))

# read in bibtex, write out RDF XML
if (interactive()) {
  (z <- system.file('extdata/bibtex.bib', package = "handlr"))
  (x <- HandlrClient$new(x = z))
  x$path
  x$format
  x$read("bibtex")
  x$parsed
  x$write("rdfxml")
  cat(x$write("rdfxml"))
}

# codemeta
(z <- system.file('extdata/codemeta.json', package = "handlr"))
(x <- HandlrClient$new(x = z))
x$path
x$format
x$read("codemeta")
x$parsed
x$write("codemeta")

# > 1
z <- system.file('extdata/citeproc-many.json', package = "handlr")
```

```
(x <- HandlrClient$new(x = z))
x$parsed
x$read()
x$parsed
## schmea org
x$write("schema_org")
## bibtex
x$write("bibtex")
## bibtex to file
f <- tempfile(fileext=".bib")
x$write("bibtex", f)
readLines(f)
unlink(f)
## to RIS
x$write("ris")
### only one per file, so not combined
files <- replicate(2, tempfile(fileext=".ris"))
x$write("ris", files)
lapply(files, readLines)

# handle strings instead of files
z <- system.file('extdata/citeproc-crossref.json', package = "handlr")
(x <- HandlrClient$new(x = readLines(z)))
x$read("citeproc")
x$parsed
cat(x$write("bibtex"), sep = "\n")
```

`handl_to_df` *handl to data.frame conversion*

Description

handl to data.frame conversion

Usage

```
handl_to_df(x)
```

Arguments

x	an object of class handl

Value

data.frame with column following handl, with as many rows as there are citations

Note

requires the Suggested package `data.table`

Examples

```
z <- system.file('extdata/crossref.ris', package = "handlr")
res <- ris_reader(z)
handl_to_df(res)

(x <- HandlrClient$new(x = z))
x$as_df() # empty data.frame
x$read()
x$as_df() # data.frame with citation data

(z <- system.file('extdata/bib-many.bib', package = "handlr"))
res2 <- bibtex_reader(x = z)
handl_to_df(res2)
```

---

rdf_xml_writer *RDF XML writer*

Description

RDF XML writer

Usage

```
rdf_xml_writer(z, ...)
```

Arguments

z	an object of class handl; see handl for more
...	further params passed to jsonld::jsonld_to_rdf()

Details

package jsonld required for this writer

Value

RDF XML

See Also

Other writers: bibtex_writer, citeproc_writer, codemeta_writer, ris_writer, schema_org_writer

Examples

```
if (requireNamespace("jsonld") && interactive()) {
  library("jsonld")
  z <- system.file('extdata/citeproc.json', package = "handlr")
  (tmp <- citeproc_reader(z))

  (z <- system.file('extdata/bibtex.bib', package = "handlr"))
  (tmp <- bibtex_reader(z))
  rdf_xml_writer(z = tmp)
  cat(rdf_xml_writer(z = tmp))
}
```

`ris_reader` *ris reader*

Description

ris reader

Usage

```
ris_reader(x)
```

Arguments

x	(character) a file path or string

Value

an object of class `handl`; see handl for more

References

RIS tags https://en.wikipedia.org/wiki/RIS_(file_format)

See Also

Other readers: `bibtex_reader`, `citeproc_reader`, `codemeta_reader`

Other ris: `ris_writer`

Examples

```
z <- system.file('extdata/crossref.ris', package = "handlr")
ris_reader(z)

z <- system.file('extdata/peerj.ris', package = "handlr")
ris_reader(z)

z <- system.file('extdata/plos.ris', package = "handlr")
```

```
ris_reader(z)

# from a string
z <- system.file('extdata/crossref.ris', package = "handlr")
my_string <- ris_writer(ris_reader(z))
class(my_string)
ris_reader(my_string)

# many
z <- system.file('extdata/multiple-eg.ris', package = "handlr")
ris_reader(z)
```

`ris_writer` *ris writer*

Description

ris writer

Usage

```
ris_writer(z)
```

Arguments

`z`	an object of class `handl`; see handl for more

Value

text if one RIS citation or list of many

References

RIS tags https://en.wikipedia.org/wiki/RIS_(file_format)

See Also

Other writers: `bibtex_writer`, `citeproc_writer`, `codemeta_writer`, `rdf_xml_writer`, `schema_org_writer`

Other ris: `ris_reader`

Examples

```
# from a RIS file
z <- system.file('extdata/crossref.ris', package = "handlr")
tmp <- ris_reader(z)
cat(ris_writer(z = tmp))

# peerj
z <- system.file('extdata/peerj.ris', package = "handlr")
```

```
tmp <- ris_reader(z)
cat(ris_writer(z = tmp))

# plos
z <- system.file('extdata/plos.ris', package = "handlr")
tmp <- ris_reader(z)
cat(ris_writer(z = tmp))

# elsevier
z <- system.file('extdata/elsevier.ris', package = "handlr")
tmp <- ris_reader(z)
cat(ris_writer(z = tmp))

z <- system.file('extdata/citeproc.json', package = "handlr")
res <- citeproc_reader(z)
cat(ris_writer(z = res))

# many
## combine many RIS in a handl object
z <- system.file('extdata/crossref.ris', package = "handlr")
cr <- ris_reader(z)
z <- system.file('extdata/peerj.ris', package = "handlr")
prj <- ris_reader(z)
c(cr, prj)

# many bibtex to ris via c method
a <- system.file('extdata/bibtex.bib', package = "handlr")
b <- system.file('extdata/crossref.bib', package = "handlr")
aa <- bibtex_reader(a)
bb <- bibtex_reader(a)
(res <- c(aa, bb))
cat(ris_writer(res), sep = "\n\n")

## manhy Citeproc to RIS
z <- system.file('extdata/citeproc-many.json', package = "handlr")
w <- citeproc_reader(x = z)
ris_writer(w)
cat(ris_writer(w), sep = "\n")
```

---

schema_org_writer *Schema org writer*

Description

Schema org writer

Usage

```
schema_org_writer(z, auto_unbox = TRUE, pretty = TRUE, ...)
```

Arguments

z	an object of class handl; see handl for more
auto_unbox	(logical) automatically "unbox" all atomic vectors of length 1 (default: TRUE). passed to jsonlite::toJSON()
pretty	(logical) adds indentation whitespace to JSON output (default: TRUE), passed to jsonlite::toJSON()
...	further params passed to jsonlite::toJSON()

Value

an object of class json

See Also

Other writers: bibtex_writer, citeproc_writer, codemeta_writer, rdf_xml_writer, ris_writer

Examples

```
(z <- system.file('extdata/bibtex.bib', package = "handlr"))
(tmp <- bibtex_reader(z))
schema_org_writer(tmp)
schema_org_writer(tmp, pretty = FALSE)

# many citeproc to schema
z <- system.file('extdata/citeproc-many.json', package = "handlr")
w <- citeproc_reader(x = z)
schema_org_writer(w)
schema_org_writer(w, pretty = FALSE)
```

Index