Package 'pubchunks'

October 14, 2022

Title Fetch Sections of XML Scholarly Articles

Description Get chunks of XML scholarly articles without having to know how to work with XML. Custom mappers for each publisher and for each article section pull out the information you want. Works with outputs from package 'fulltext', 'xml2' package documents, and file paths to XML documents.

Version 0.3.0

License MIT + file LICENSE

URL https://docs.ropensci.org/pubchunks/, https://github.com/ropensci/pubchunks

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

Encoding UTF-8

Language en-US

Imports xml2 (>= 1.1.1), data.table, rcrossref

Suggests fulltext (>= 1.0.1), testthat

RoxygenNote 7.1.1

X-schema.org-applicationCategory Literature

X-schema.org-keywords text-ming, literature, pdf, xml, publications, citations, full-text, TDM

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

NeedsCompilation no

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

Maintainer Scott Chamberlain <sckott@protonmail.com>

Repository CRAN

Date/Publication 2020-09-04 15:20:02 UTC

R topics documented:

---

pubchunks-package *Get chunks of XML articles*

Description

Pass XML in various forms, and get out specific sections of articles. It makes not knowing what XPath/CSS selectors are totally fine.

Author(s)

Scott Chamberlain

---

pub_chunks *Extract chunks of data from articles*

Description

`pub_chunks` makes it easy to extract sections of an article. You can extract just authors across all articles, or all references sections, or the complete text of each article. Then you can pass the output downstream for visualization and analysis.

Usage

```
pub_chunks(x, sections = "all", provider = NULL)
```

Arguments

`x`	one of the following: • file path for an XML file • a character string of XML, a list (of file paths, or XML in a character string, or `xml_document` objects) • or an object of class `fulltext::ft_data`, the output from a call to `fulltext::ft_get()`
`sections`	(character) What elements to get, can be one or more in a vector or list. See `pub_sections()` for options. optional. Default is to get all sections. See Details.

provider (character) a single publisher name. see pub_providers() for options. required. If you select the wrong provider for the XML you have you may or may not get what you need :). By default this is NULL and we use pub_guess_publisher() to guess the publisher; we may get it wrong. You can override our guessing by passing in a name.

Details

Options for the sections parameter:

- front - Publisher, journal and article metadata elements
- body - Body of the article
- back - Back of the article, acknowledgments, author contributions, references
- title - Article title
- doi - Article DOI
- categories - Publisher's categories, if any
- authors - Authors
- aff - Affiliation (includes author names)
- keywords - Keywords
- abstract - Article abstract
- executive_summary - Article executive summary
- refs - References
- refs_dois - References DOIs - if available
- publisher - Publisher name
- journal_meta - Journal metadata
- article_meta - Article metadata
- acknowledgments - Acknowledgments
- permissions - Article permissions
- history - Dates, recieved, published, accepted, etc.

Value

A list, named by the section selected. sections not found or not in accepted list return NULL or zero length list. A ".publisher" list element gets attached to each list output, even when no data is found. When fulltext::ft_get output is passed in here, the list is named by the publisher, then within each publisher is a list of articles named by their identifiers (e.g. DOIs).

Examples

```
# a file path to an XML file
x <- system.file("examples/elsevier_1.xml", package = "pubchunks")
pub_chunks(x, "title")
pub_chunks(x, "authors")
pub_chunks(x, "acknowledgments")
```

```
pub_chunks(x, "refs")
pub_chunks(x, c("title", "refs"))

## Not run:
# works the same with the xml already in a string
xml <- paste0(readLines(x), collapse = "")
pub_chunks(xml, "title")

# also works if you've already read in the XML (with xml2 pkg)
xml <- paste0(readLines(x), collapse = "")
xml <- xml2::read_xml(xml)
pub_chunks(xml, "title")

# Hindawi
x <- system.file("examples/hindawi_1.xml", package = "pubchunks")
pub_chunks(x, "abstract")
pub_chunks(x, "authors")
pub_chunks(x, "aff")
pub_chunks(x, "title")
pub_chunks(x, "refs")$refs
pub_chunks(x, c("abstract", "title", "authors", "refs"))

# Pensoft
x <- system.file("examples/pensoft_1.xml", package = "pubchunks")
pub_chunks(x, "abstract")
pub_chunks(x, "aff")
pub_chunks(x, "title")
pub_chunks(x, "refs")$refs
pub_chunks(x, c("abstract", "title", "authors", "refs"))

# Peerj
x <- system.file("examples/peerj_1.xml", package = "pubchunks")
pub_chunks(x, "abstract")
pub_chunks(x, "authors")
pub_chunks(x, "aff")
pub_chunks(x, "title")
pub_chunks(x, "refs")$refs
pub_chunks(x, c("abstract", "title", "authors", "refs"))

# Frontiers
x <- system.file("examples/frontiers_1.xml", package = "pubchunks")
pub_chunks(x, "authors")
pub_chunks(x, "aff")
pub_chunks(x, "refs")$refs
pub_chunks(x, c("doi", "abstract", "title", "authors", "refs", "abstract"))

# eLife
x <- system.file("examples/elife_1.xml", package = "pubchunks")
pub_chunks(x, "authors")
pub_chunks(x, "aff")
pub_chunks(x, "refs")$refs
pub_chunks(x, c("doi", "title", "authors", "refs"))
```

```
# f1000research
x <- system.file("examples/f1000research_3.xml", package = "pubchunks")
pub_chunks(x, "title")
pub_chunks(x, "aff")
pub_chunks(x, "refs")$refs
pub_chunks(x, c("doi", "title", "authors", "keywords", "refs"))

# Copernicus
x <- system.file("examples/copernicus_1.xml", package = "pubchunks")
pub_chunks(x, c("doi", "abstract", "title", "authors", "refs"))
pub_chunks(x, "aff")
pub_chunks(x, "refs")$refs

# MDPI
x <- system.file("examples/mdpi_1.xml", package = "pubchunks")
x <- system.file("examples/mdpi_2.xml", package = "pubchunks")
pub_chunks(x, "title")
pub_chunks(x, "aff")
pub_chunks(x, "refs")$refs
vv <- pub_chunks(x, c("doi", "title", "authors", "keywords", "refs",
  "abstract", "categories"))
vv$doi
vv$title
vv$authors
vv$keywords
vv$refs
vv$abstract
vv$categories

# Many inputs at once
x <- system.file("examples/frontiers_1.xml", package = "pubchunks")
y <- system.file("examples/elife_1.xml", package = "pubchunks")
z <- system.file("examples/f1000research_1.xml", package = "pubchunks")
pub_chunks(list(x, y, z), c("doi", "title", "authors", "refs"))

# non-XML files/content are xxx?
# pub_chunks('foo bar')

# Pubmed brief XML files (abstract only)
x <- system.file("examples/pubmed_brief_1.xml", package = "pubchunks")
pub_chunks(x, "title")

# Pubmed full XML files
x <- system.file("examples/pubmed_full_1.xml", package = "pubchunks")
pub_chunks(x, "title")

# using output of fulltext::ft_get()
if (requireNamespace("fulltext", quietly = TRUE)) {
  library("fulltext")

  # single
  x <- fulltext::ft_get('10.7554/eLife.03032')
  pub_chunks(fulltext::ft_collect(x), sections="authors")
```

```
  # many
  dois <- c('10.1371/journal.pone.0086169', '10.1371/journal.pone.0155491',
    '10.7554/eLife.03032')
  x <- fulltext::ft_get(dois)
  pub_chunks(fulltext::ft_collect(x), sections="authors")

  # as.ft_data() function
  x <- ft_collect(as.ft_data())
  names(x)
  x$cached
  pub_chunks(x, "title")
  pub_chunks(x, "title") %>% pub_tabularize()
}

## End(Not run)
```

---

pub_guess_publisher *Guess the publisher from an XML document*

Description

Guess the publisher from an XML document

Usage

```
pub_guess_publisher(x)
```

Arguments

x	an XML file, a character string of XML, or a `xml_document` object (as from `xml2::read_xml`)

Value

a list, with two named character strings, one for `full_name` and the other a `short_name`

Examples

```
## Not run:
(x <- system.file("examples/pensoft_1.xml", package = "pubchunks"))
pub_guess_publisher(x)

(x <- system.file("examples/copernicus_2.xml", package = "pubchunks"))
pub_guess_publisher(x)

(x <- system.file("examples/peerj_1.xml", package = "pubchunks"))
pub_guess_publisher(x)

(x <- system.file("examples/hindawi_1.xml", package = "pubchunks"))
```

```
pub_guess_publisher(x)

(x <- system.file("examples/frontiers_1.xml", package = "pubchunks"))
pub_guess_publisher(x)

(x <- system.file("examples/elife_1.xml", package = "pubchunks"))
pub_guess_publisher(x)

(x <- system.file("examples/elsevier_1.xml", package = "pubchunks"))
pub_guess_publisher(x)

x <- system.file("examples/f1000research_1.xml", package = "pubchunks")
pub_guess_publisher(x)

x <- system.file("examples/plos_1.xml", package = "pubchunks")
pub_guess_publisher(x)

x <- system.file("examples/mdpi_1.xml", package = "pubchunks")
pub_guess_publisher(x)

x <- system.file("examples/pubmed_brief_1.xml", package = "pubchunks")
pub_guess_publisher(x)

x <- system.file("examples/pubmed_full_1.xml", package = "pubchunks")
pub_guess_publisher(x)

x <- system.file("examples/pubmed_full_2.xml", package = "pubchunks")
pub_guess_publisher(x)

x <- system.file("examples/pubmed_full_3.xml", package = "pubchunks")
pub_guess_publisher(x)

## End(Not run)
```

`pub_providers` *Providers*

Description

The possible providers to select from

Usage

```
pub_providers()
```

Value

character vector

Examples

```
pub_providers()
```

`pub_sections` *Sections*

Description

The possible sections of an XML article that are supported for retrieval

Usage

```
pub_sections()
```

Value

character vector

Examples

```
pub_sections()
```

`pub_tabularize` *Tabularize chunks output*

Description

Tabularize chunks output

Usage

```
pub_tabularize(x, bind = FALSE)
```

Arguments

`x`	the output of `pub_chunks()`
`bind`	(logical) whether to bind list of data.frames or not. ignored unless `list` input to `x`. default: `FALSE`

Value

a data.frame or list

Examples

```
## Not run:
# one at a time
## example 1, a file path
x <- system.file("examples/elife_1.xml", package = "pubchunks")
(res <- pub_chunks(x, c("doi", "title", "keywords")))
pub_tabularize(res)

## example 2, a file path
y <- system.file("examples/frontiers_1.xml", package = "pubchunks")
(res2 <- pub_chunks(y, c("doi", "title", "keywords")))
pub_tabularize(res2)

# > 1, a list of file paths
x <- system.file("examples/elife_1.xml", package = "pubchunks")
y <- system.file("examples/frontiers_1.xml", package = "pubchunks")
(res <- pub_chunks(list(x, y), c("doi", "title", "keywords")))
pub_tabularize(res)
pub_tabularize(res, bind = TRUE)

# using output of fulltext::ft_get()
if (requireNamespace("fulltext", quietly = TRUE)) {
  dois <- c('10.1371/journal.pone.0086169', '10.1371/journal.pone.0155491',
    '10.7554/eLife.03032')
  x <- fulltext::ft_get(dois)
  (tmp <- pub_chunks(fulltext::ft_collect(x), sections=c("doi","title")))
  pub_tabularize(tmp)
  pub_tabularize(tmp, bind = TRUE)
}
## End(Not run)
```

Index