Package 'OAIHarvester'

January 20, 2025

Version 0.3-5

Title Harvest Metadata Using OAI-PMH Version 2.0

Description Harvest metadata using the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) version 2.0 (for more information, see <https://www.openarchives.org/OAI/openarchivesprotocol.html>).

Imports utils, curl, xml2

License GPL-2

NeedsCompilation no

Author Kurt Hornik [aut, cre] (<https://orcid.org/0000-0003-4198-9911>)

Maintainer Kurt Hornik <Kurt.Hornik@R-project.org>

Repository CRAN

Date/Publication 2024-11-13 07:13:16 UTC

Contents

`harvest` *OAI-PMH Harvester*

Description

Harvest a repository using Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) requests.

Usage

```
oaih_harvest(baseurl, prefix = "oai_dc",
             from = NULL, until = NULL, set = NULL,
             transform = TRUE)
```

Arguments

baseurl	a character string giving the base URL of the repository.
prefix	a character vector with the formats in which metadata should be obtained, or NULL, indicating all available formats. The default ("oai_dc") corresponds to the mandatory OAI unqualified Dublin Core metadata schema.
from, until	character strings or Date or POSIXt date/time objects giving datestamps to be used as lower or upper bounds, respectively, for datestamp-based selective harvesting (i.e., only harvest records with datestamps in the given range). If character, dates and times must be encoded using ISO 8601 in either '%F' or '%FT%TZ' format (see strptime). The trailing 'Z' must be used when including time. OAI-PMH implies UTC for data/time specifications.
set	a character vector giving the sets to be used for selective harvesting (i.e., only harvest records in the given sets), or NULL.
transform	a logical indicating whether the OAI-PMH XML results to "useful" R data structures via oaih_transform. Default: true.

Details

This is a high-level function for conveniently harvesting metadata from a repository, allowing specifying several metadata formats or sets. It also maps datestamps specified as R date or date/time objects to valid OAI-PMH datestamps according to the granularity of the repository.

Value

If the OAI-PMH request was successful, the result of the request as XML or (default) transformed to "useful" R data structures.

---

providers *OAI-PMH Providers*

Description

Names, base URLs and identifiers of registered and validated OAI conforming metadata providers.

Usage

```
oaih_providers()
```

Details

Information is extracted from https://www.openarchives.org/Register/BrowseSites (as the XML formatted list of base URLs of registered data providers from https://www.openarchives.org/pmh/registry/ListFriends does not provide repository names), and cached for the current R session.

Value

A character data frame with variables `name`, `baseurl` and `identifier` providing the repository names, base URLs and OAI identifier (see https://www.openarchives.org/OAI/2.0/guidelines-oai-identifier.htm).

`serialize` *Serialization for OAI-PMH Objects*

Description

Functions to write a single OAI-PMH object to a file, and to restore it, and to perform the necessary conversions of XML objects to and from strings.

Usage

```
oaih_read_RDS(file, ...)
oaih_save_RDS(x, ...)
oaih_str_to_xml(x)
oaih_xml_to_str(x)
```

Arguments

`x`	an R object.
`file`	a connection or the name of the file where the R object is saved to.
`...`	arguments to be passed to `readRDS` (`oaih_read_RDS`) `saveRDS` (`oaih_save_RDS`).

Details

The OAI-PMH objects obtained by OAI-PMH requests (e.g., `oaih_list_records`) and subsequent transformations (`oaih_transform`) are made up of both character vectors and XML nodes from package **xml2**, with the latter lists of external pointers. Thus, serialization does not work "out of the box", and in fact using refhooks in calls to `readRDS` or `saveRDS` does not work either (as one needs to (de)serialize a list of pointers, and not a single one). We thus provide helper functions to (recursively) (de)serialize the XML objects to/from strings, and to pre-process R objects before saving to a file and post-process after restoring from a file.

Examples

```
tryCatch({
## Run inside tryCatch() so that checks fail gracefully if OAI-PMH
## requests time out or fail otherwise.
baseurl <- "https://research.wu.ac.at/ws/oai"
x <- oaih_identify(baseurl)
## Now 'x' is a list of character vectors and XML nodes:
x
## To save to a file and restore:
f <- tempfile()
oaih_save_RDS(x, file = f)
y <- oaih_read_RDS(f)
all.equal(x, y)
## Equivalently, we can directly pre-process before saving and
## post-process after restoring:
saveRDS(oaih_xml_to_str(x), f)
z <- oaih_str_to_xml(readRDS(f))
all.equal(y, z)
##
}, error = identity)
```

---

size *OAI-PMH Repository Size*

Description

Determine the number of items available for (selective) harvesting in an OAI repository.

Usage

```
oaih_size(baseurl, from = NULL, until = NULL, set = NULL)
```

Arguments

`baseurl`	a character string giving the base URL of the repository.
`from`, `until`	character strings or Date or POSIXt date/time objects giving datestamps to be used as lower or upper bounds, respectively, for datestamp-based selective harvesting (i.e., only consider records with datestamps in the given range). If character, dates and times must be encoded using ISO 8601 in either '`%F`' or '`%FT%TZ`' format (see `strptime`). The trailing '`Z`' must be used when including time. OAI-PMH implies UTC for data/time specifications.
`set`	a character vector giving the sets to be considered for selective harvesting (i.e., only consider records in the given sets), or `NULL`.

Details

Determining the number of items without actually harvesting these is only possible if the repository's flow control mechanism provides `resumptionToken` elements with `completeListSize` attributes (see https://www.openarchives.org/OAI/openarchivesprotocol.html), or flow control is not applied when listing identifiers in the selected range.

Value

A numeric giving the number of items available for (selective) harvesting, or `NA_real_` if the number could not be determined without harvesting.

Examples

```
tryCatch({
## Run inside tryCatch() so that checks fail gracefully if OAI-PMH
## requests time out or fail otherwise.
oaih_size("https://www.jstatsoft.org/oai")
##
}, error = identity)
```

`transform` *Transform OAI-PMH XML Results*

Description

Transform OAI-PMH XML results to "useful" R data structures (lists of character vectors or XML nodes) for further processing or analysis.

Usage

```
oaih_transform(x)
```

Arguments

`x`	an XML node, or a list of character vectors or XML nodes.

Details

In a "list context", i.e., if `x` conceptually contains information on several cases, transformation gives a "list matrix" (a list of character vector or XML node observations with a dim attribute) providing a rectangular case by variables data layout; otherwise, a list of variables. See the vignette for details.

Value

A list of character vectors or XML nodes, arranged as a matrix in the "list context".

Examples

```
tryCatch({
## Run inside tryCatch() so that checks fail gracefully if OAI-PMH
## requests time out or fail otherwise.
baseurl <- "https://research.wu.ac.at/ws/oai"
## Get a single record to save bandwidth.
x <- oaih_get_record(baseurl,
              "oai:research.wu.ac.at:publications/783bfc47-bf51-454d-8b78-33fd63243e48",
                     transform = FALSE)
## The result of the request is a single OAI-PMH XML <record> node:
x
## Transform this (turning identifier, datestamp and setSpec into
## character data):
x <- oaih_transform(x)
x
## This has its metadata in the default Dublin Core form, encoded in
## XML.  Transform these to character data:
oaih_transform(x$metadata)
##
}, error = identity)
```

---

verb *OAI-PMH Verb Functions*

Description

Perform Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) requests for harvesting repositories.

Usage

```
oaih_get_record(baseurl, identifier, prefix = "oai_dc",
                transform = TRUE)
oaih_identify(baseurl, transform = TRUE)
oaih_list_identifiers(baseurl, prefix = "oai_dc", from = NULL,
                      until = NULL, set = NULL, transform = TRUE)
oaih_list_metadata_formats(baseurl, identifier = NULL,
                           transform = TRUE)
oaih_list_records(baseurl, prefix = "oai_dc", from = NULL,
                  until = NULL, set = NULL, transform = TRUE)
oaih_list_sets(baseurl, transform = TRUE)
```

Arguments

baseurl	a character string giving the base URL of the repository.
identifier	a character string giving the unique identifier for an item in a repository.

prefix	a character string to specify the metadata format in OAI-PMH requests issued to the repository. The default ("oai_dc") corresponds to the mandatory OAI unqualified Dublin Core metadata schema.
from, until	character strings giving datestamps to be used as lower or upper bounds, respectively, for datestamp-based selective harvesting (i.e., only harvest records with datestamps in the given range). Dates and times must be encoded using ISO 8601 in either '%F' or '%FT%TZ' format (see strptime). The trailing 'Z' must be used when including time. OAI-PMH implies UTC for data/time specifications.
set	a character string giving a set to be used for selective harvesting (i.e., only harvest records in the given set).
transform	a logical indicating whether the OAI-PMH XML results to "useful" R data structures via oaih_transform. Default: true.

Value

If the OAI-PMH request was successful, the result of the request as XML or (default) transformed to "useful" R data structures.

Examples

```
tryCatch({
## Run inside tryCatch() so that checks fail gracefully if OAI-PMH
## requests time out or fail otherwise.
##
## Harvest WU Reearch metadata.
baseurl <- "https://research.wu.ac.at/ws/oai"
## Identify.
oaih_identify(baseurl)
## List metadata formats.
oaih_list_metadata_formats(baseurl)
## List sets.
sets <- oaih_list_sets(baseurl)
head(sets, 20L)
## List records in the 'year 1986' set.
spec <- "publications:year1986"
x <- oaih_list_records(baseurl, set = spec)
## Extract the metadata.
m <- x[, "metadata"]
m <- oaih_transform(m[lengths(m) > 0L])
## Find the most frequent keywords.
keywords <- unlist(m[, "subject"])
keywords <- keywords[!startsWith(keywords, "/dk/atira/pure")]
head(sort(table(keywords), decreasing = TRUE))
##
}, error = identity)
```

Index