Package ‘geonapi’

March 21, 2024

Type Package

Title 'GeoNetwork' API R Interface

Version 0.7-2

Date 2024-03-21

Maintainer Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Description Provides an R interface to the 'GeoNetwork' API (<https://geonetwork-opensource.org/#api>) allowing to upload and publish metadata in a 'GeoNetwork' web-application and expose it to OGC CSW.

Depends R (>= 3.1.0), geometa, keyring

Imports R6, openssl, httr, XML, plyr

Suggests testthat, roxygen2

License MIT + file LICENSE

URL https://github.com/eblondel/geonapi/wiki,
https://geonetwork-opensource.org

BugReports https://github.com/eblondel/geonapi/issues

LazyLoad yes

RoxygenNote 7.2.3

NeedsCompilation no

Author Emmanuel Blondel [aut, cre] (<https://orcid.org/0000-0002-5870-5762>)

Repository CRAN

Date/Publication 2024-03-21 21:40:02 UTC

R topics documented:

`geonapi` *'GeoNetwork' API R Interface*

Description

Provides an R interface to the 'GeoNetwork' API (<https://geonetwork-opensource.org/#api>) allowing to upload and publish metadata in a 'GeoNetwork' web-application and expose it to OGC CSW Web-Services (Catalogue Service for the Web).

Author(s)

Emmanuel Blondel <`emmanuel.blondel1@gmail.com`>

`GNAbstractManager` *GNAbstractManager*

Description

GNAbstractManager

GNAbstractManager

Format

`R6Class` object.

Value

Object of `R6Class` with methods for communication with the REST API of a GeoNetwork instance.

Public fields

`verbose.info` If package info log messages have to be printed out

`verbose.debug` If curl debug log messages have to be printed out

`loggerType` the type of logger

`url` the Base url of GeoNetwork

`version` the version of GeoNetwork. Handled as `GNVersion` object

`lang` the language for Geonetwork service. Default is `eng`

`basicAuth` if basic auth is performed

Methods

Public methods:

- `GNAbstractManager$logger()`
- `GNAbstractManager$INFO()`
- `GNAbstractManager$WARN()`
- `GNAbstractManager$ERROR()`
- `GNAbstractManager$new()`
- `GNAbstractManager$getUrl()`
- `GNAbstractManager$getLang()`
- `GNAbstractManager$login()`
- `GNAbstractManager$getClassName()`
- `GNAbstractManager$clone()`

Method `logger()`**:** Provides log messages

Usage:

```
GNAbstractManager$logger(type, text)
```

Arguments:

`type` type of log ("INFO", "WARN", "ERROR")

`text` the log message text

Method `INFO()`**:** Provides INFO log messages

Usage:

```
GNAbstractManager$INFO(text)
```

Arguments:

`text` the log message text

Method `WARN()`**:** Provides WARN log messages

Usage:

```
GNAbstractManager$WARN(text)
```

Arguments:

`text` the log message text

Method `ERROR()`**:** Provides ERROR log messages

Usage:

```
GNAbstractManager$ERROR(text)
```

Arguments:

`text` the log message text

Method `new()`**:** This method is used to instantiate a GNAbstractManager with the `url` of the GeoNetwork and credentials to authenticate (`user`/`pwd`). By default, the `logger` argument will be set to `NULL` (no logger).

The `keyring_backend` can be set to use a different backend for storing the Geonetwork password/token with **keyring** (Default value is 'env').

The logger can be either NULL, "INFO" (with minimum logs), or "DEBUG" (for complete curl http calls logs)

Usage:

```
GNAbstractManager$new(
  url,
  user = NULL,
  pwd = NULL,
  version,
  logger = NULL,
  keyring_backend = "env"
)
```

Arguments:

`url` url

`user` user

`pwd` pwd

`version` version

`logger` logger

`keyring_backend` keyring backend. Default is 'env'

Method `getUrl()`**:** Get URL

Usage:

```
GNAbstractManager$getUrl()
```

Returns: an object of class `character`

Method `getLang()`**:** Get service language

Usage:

```
GNAbstractManager$getLang()
```

Returns: an object of class `character`

Method `login()`**:** Log-ins. This methods (here abstract) attempts a connection to GeoNetwork API. Used internally by subclasses of GNAbstractManager to login Geonetwork.

Usage:

```
GNAbstractManager$login(user, pwd)
```

Arguments:

`user` user

`pwd` pwd

Method `getClassName()`**:** Get class name

Usage:

```
GNAbstractManager$getClassName()
```

Returns: an object of class `character`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNAbstractManager$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

GNLegacyAPIManager *GNLegacyAPIManager*

Description

GNLegacyAPIManager

GNLegacyAPIManager

Format

R6Class object.

Value

Object of R6Class with methods for communication with the REST API of a GeoNetwork instance using the legacy API.

Super class

geonapi::GNAbstractManager -> GNLegacyAPIManager

Methods

Public methods:

- GNLegacyAPIManager$new()
- GNLegacyAPIManager$login()
- GNLegacyAPIManager$getGroups()
- GNLegacyAPIManager$getCategories()
- GNLegacyAPIManager$insertMetadata()
- GNLegacyAPIManager$setPrivConfiguration()
- GNLegacyAPIManager$get()
- GNLegacyAPIManager$getMetadataByID()
- GNLegacyAPIManager$getMetadataByUUID()
- GNLegacyAPIManager$getInfoByID()
- GNLegacyAPIManager$getInfoByUUID()
- GNLegacyAPIManager$updateMetadata()
- GNLegacyAPIManager$deleteMetadata()
- GNLegacyAPIManager$deleteMetadataAll()
- GNLegacyAPIManager$clone()

Method `new()`**:** This method is used to instantiate a GNLegacyAPIManager with the `url` of the GeoNetwork and credentials to authenticate (`user`/`pwd`).

The `keyring_backend` can be set to use a different backend for storing the Geonetwork password/token with **keyring** (Default value is 'env').

The logger can be either NULL, "INFO" (with minimum logs), or "DEBUG" (for complete curl http calls logs)

Usage:

```
GNLegacyAPIManager$new(
  url,
  user = NULL,
  pwd = NULL,
  version,
  logger = NULL,
  keyring_backend = "env"
)
```

Arguments:

`url` url

`user` user

`pwd` pwd

`version` version

`logger` logger

`keyring_backend` keyring backend. Default is 'env'

Method `login()`**:** #' This methods attempts a connection to GeoNetwork REST API. User internally during initialization of `GNLegacyAPIManager`.

Usage:

```
GNLegacyAPIManager$login(user, pwd)
```

Arguments:

`user` user

`pwd` pwd

Method `getGroups()`**:** Retrieves the list of user groups available in Geonetwork

Usage:

```
GNLegacyAPIManager$getGroups()
```

Returns: an object of class `data.frame`

Method `getCategories()`**:** Retrieves the list of categories available in Geonetwork

Usage:

```
GNLegacyAPIManager$getCategories()
```

Returns: an object of class `data.frame`

Method `insertMetadata()`**:** Inserts a metadata by file, XML object or **geometa** object of class `ISOMetadata` or `ISOFeatureCatalogue`. If successful, returns the Geonetwork metadata internal identifier (integer). Extra parameters `geometa_validate` (TRUE by default) and `geometa_inspire` (FALSE by default) can be used with geometa objects for perform ISO and INSPIRE validation respectively. In that case on object of class `geometa::INSPIREMetadataValidator`, with a proper user API key, should be specified as `geometa_inspireValidator` argument.

Usage:

```
GNLegacyAPIManager$insertMetadata(
  xml = NULL,
  file = NULL,
  geometa = NULL,
  group,
  category = NULL,
  stylesheet = NULL,
  validate = FALSE,
  geometa_validate = TRUE,
  geometa_inspire = FALSE,
  geometa_inspireValidator = NULL
)
```

Arguments:

`xml` xml object of class XMLInternalNode-class from **XML**

`file` file

`geometa` geometa, object of class `ISOMetadata` or `ISOFeatureCatalogue` from **geometa**

`group` group

`category` category

`stylesheet` stylesheet

`validate` validate

`geometa_validate` validate geometa object

`geometa_inspire` validate geometa object vs. INSPIRE

`geometa_inspireValidator` geometa INSPIRE validator to use

Method `setPrivConfiguration():` Set the privilege configuration for a metadata. 'id' is the metadata integer id. 'config' is an object of class "GNPrivConfiguration".

Usage:

```
GNLegacyAPIManager$setPrivConfiguration(id, config)
```

Arguments:

`id` id

`config` config

Method `get():` Generic getter for metadata. Possible values for by are 'id', 'uuid'. Used internally only. The 'output' argument gives the type of output to return, with possible values "id", "metadata", "info".

Usage:

```
GNLegacyAPIManager$get(id, by, output)
```

Arguments:

`id` id

`by` by

`output` output

Method `getMetadataByID():` Get a metadata by Id

Usage:

```
GNLegacyAPIManager$getMetadataByID(id)
```

Arguments:

`id` id

Returns: an object of class `ISOMetadata` (ISO 19115) or `ISOFeatureCatalogue` (ISO 19110) (from **geometa** package)

Method `getMetadataByUUID()`**:** Get a metadata by UUID

Usage:

```
GNLegacyAPIManager$getMetadataByUUID(uuid)
```

Arguments:

`uuid` uuid

Returns: an object of class `ISOMetadata` (ISO 19115) or `ISOFeatureCatalogue` (ISO 19110) (from **geometa** package)

Method `getInfoByID()`**:** Get a metadata Info by Id.

Usage:

```
GNLegacyAPIManager$getInfoByID(id)
```

Arguments:

`id` id

Returns: an XML document object

Method `getInfoByUUID()`**:** Get a metadata Info by UUID

Usage:

```
GNLegacyAPIManager$getInfoByUUID(uuid)
```

Arguments:

`uuid` uuid

Returns: an XML document object

Method `updateMetadata()`**:** Updates a metadata by file, XML object or **geometa** object of class 'ISOMetadata' or 'ISOFeatureCatalogue'. Extra parameters `geometa_validate` (TRUE by default) and `geometa_inspire` (FALSE by default) can be used with geometa objects for perform ISO and INSPIRE validation respectively. In that case on object of class `geometa::INSPIREMetadataValidator`, with a proper user API key, should be specified as `geometa_inspireValidator` argument.

Usage:

```
GNLegacyAPIManager$updateMetadata(
  id,
  xml = NULL,
  file = NULL,
  geometa = NULL,
  geometa_validate = TRUE,
  geometa_inspire = FALSE,
  geometa_inspireValidator = NULL
)
```

Arguments:

`id` metadata id

`xml` xml object of class XMLInternalNode-class from **XML**

`file` file

`geometa` geometa, object of class `ISOMetadata` or `ISOFeatureCatalogue` from **geometa**

`geometa_validate` validate geometa object

`geometa_inspire` validate geometa object vs. INSPIRE

`geometa_inspireValidator` geometa INSPIRE validator to use

Method `deleteMetadata()`**:** Deletes metadata by Id.

Usage:

```
GNLegacyAPIManager$deleteMetadata(id)
```

Arguments:

`id` id

Returns: the id of the record deleted, `NULL` otherwise

Method `deleteMetadataAll()`**:** Deletes all metadata

Usage:

```
GNLegacyAPIManager$deleteMetadataAll()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNLegacyAPIManager$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
## Not run: 
  GNLegacyAPIManager$new("http://localhost:8080/geonetwork", "admin", "geonetwork", "3.0.0")

## End(Not run)
```

GNManager *GeoNetwork REST API Manager*

Description

The function GNManager$new will set-up the right Geonetwork manager depending on the GeoNetwork version specified by the user. For the time-being, GeoNetwork with version < 4 will be interfaced with the GeoNetwork legacy API (see detailed documentation at GNLegacyAPIManager), while starting with GeoNetwork 3.2, the new GeoNetwork OpenAPI will be used.

Format

R6Class object.

Value

Object of R6Class with methods for communication with the API of a GeoNetwork instance.

Super class

geonapi::GNAbstractManager -> GNManager

Methods

Public methods:

- GNManager$new()
- GNManager$clone()

Method new()**:** Initializes a GNManager

Usage:

```
GNManager$new(url, user = NULL, pwd = NULL, version, logger = NULL)
```

Arguments:

url url

user user

pwd pwd

version version

logger logger

Method clone()**:** The objects of this class are cloneable with this method.

Usage:

```
GNManager$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
## Not run:
   GMManager$new("http://localhost:8080/geonetwork", "admin", "geonetwork", "3.0.0")

## End(Not run)
```

`GNOpenAPIManager` *GNOpenAPIManager*

Description

GNOpenAPIManager

GNOpenAPIManager

Format

`R6Class` object.

Value

Object of `R6Class` with methods for communication with the REST API of a GeoNetwork instance using the legacy API.

Super class

`geonapi::GNAbstractManager` -> `GNOpenAPIManager`

Methods

Public methods:

- `GNOpenAPIManager$new()`
- `GNOpenAPIManager$login()`
- `GNOpenAPIManager$getGroups()`
- `GNOpenAPIManager$getTags()`
- `GNOpenAPIManager$getCategories()`
- `GNOpenAPIManager$getMetadataByUUID()`
- `GNOpenAPIManager$insertRecord()`
- `GNOpenAPIManager$insertMetadata()`
- `GNOpenAPIManager$updateMetadata()`
- `GNOpenAPIManager$deleteMetadata()`

- GNOpenAPIManager$uploadAttachment()
- GNOpenAPIManager$publishThumbnail()
- GNOpenAPIManager$doiCheckPreConditions()
- GNOpenAPIManager$createDOI()
- GNOpenAPIManager$deleteDOI()
- GNOpenAPIManager$clone()

Method `new()`**:** This method is used to instantiate a `GNOpenAPIManager` with the `url` of the GeoNetwork and credentials to authenticate (`user`/`pwd`).

The `keyring_backend` can be set to use a different backend for storing the Geonetwork password/token with **keyring** (Default value is 'env').

The logger can be either NULL, "INFO" (with minimum logs), or "DEBUG" (for complete curl http calls logs)

Usage:

```
GNOpenAPIManager$new(
  url,
  user = NULL,
  pwd = NULL,
  version,
  logger = NULL,
  keyring_backend = "env"
)
```

Arguments:

`url` url

`user` user

`pwd` pwd

`version` version

`logger` logger

`keyring_backend` keyring backend

Method `login()`**:** This methods attempts a connection to GeoNetwork REST API. User internally during initialization of `GNLegacyAPIManager`.

Usage:

```
GNOpenAPIManager$login(user, pwd)
```

Arguments:

`user` user

`pwd` pwd

Method `getGroups()`**:** Retrieves the list of user groups available in Geonetwork

Usage:

```
GNOpenAPIManager$getGroups()
```

Returns: an object of class `data.frame`

Method `getTags()`**:** Retrieves the list of tags (categories) available in Geonetwork

Usage:

```
GNOpenAPIManager$getTags()
```

Returns: an object of class `data.frame`

Method `getCategories()`**:** Retrieves the list of categories (same as tags) available in Geonetwork

Usage:

```
GNOpenAPIManager$getCategories()
```

Returns: an object of class `data.frame`

Method `getMetadataByUUID()`**:** Get a metadata by UUID.

Usage:

```
GNOpenAPIManager$getMetadataByUUID(
  uuid,
  addSchemaLocation = TRUE,
  increasePopularity = TRUE,
  approved = TRUE
)
```

Arguments:

`uuid` uuid

`addSchemaLocation` add schema location. Default is `TRUE`

`increasePopularity` increase popularity. Default is `TRUE`

`approved` approved

Returns: Returns an object of class `ISOMetadata` (ISO 19115) or `ISOFeatureCatalogue` (ISO 19110) (from **geometa** package)

Method `insertRecord()`**:** Inserts a record by file, XML object or **geometa** object of class `ISOMetadata` or `ISOFeatureCatalogue`. Extra parameters related to **geometa** objects: `geometa_validate` (TRUE by default) and `geometa_inspire` (FALSE by default) can be used to perform ISO and INSPIRE validation respectively. In that case on object of class `geometa::INSPIREMetadataValidator`, with a proper user API key, should be specified as `geometa_inspireValidator` argument.

Usage:

```
GNOpenAPIManager$insertRecord(
  xml = NULL,
  file = NULL,
  geometa = NULL,
  metadataType = "METADATA",
  uuidProcessing = "NOTHING",
  group,
  category = NULL,
  rejectIfInvalid = FALSE,
  publishToAll = TRUE,
  transformWith = "_none_",
  schema = NULL,
  extra = NULL,
  geometa_validate = TRUE,
  geometa_inspire = FALSE,
  geometa_inspireValidator = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`file` file

`geometa` geometa object of class `ISOMetadata` or `ISOFeatureCatalogue`

`metadataType` metadata type. By default `METADATA`

`uuidProcessing` UUID processing. By default `NOTHING`. Other possible value: `OVERWRITE`

`group` group

`category` category

`rejectIfInvalid` reject if invalid. Default `FALSE`

`publishToAll` publish to all. Default `TRUE`

`transformWith` transform with. Default is _none_

`schema` schema

`extra` extra

`geometa_validate` validate geometa object

`geometa_inspire` validate geometa object vs. INSPIRE

`geometa_inspireValidator` geometa INSPIRE validator to use

Method `insertMetadata()`**:** Inserts a metadata by file, XML object or **geometa** object of class `ISOMetadata` or `ISOFeatureCatalogue`. Extra parameters related to **geometa** objects: `geometa_validate` (TRUE by default) and `geometa_inspire` (FALSE by default) can be used to perform ISO and INSPIRE validation respectively. In that case on object of class `geometa::INSPIREMetadataValidator`, with a proper user API key, should be specified as `geometa_inspireValidator` argument.

Usage:

```
GNOpenAPIManager$insertMetadata(
  xml = NULL,
  file = NULL,
  geometa = NULL,
  metadataType = "METADATA",
  uuidProcessing = "NOTHING",
  group,
  category = NULL,
  rejectIfInvalid = FALSE,
  publishToAll = TRUE,
  transformWith = "_none_",
  schema = NULL,
  extra = NULL,
  geometa_validate = TRUE,
  geometa_inspire = FALSE,
  geometa_inspireValidator = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`file` file

`geometa` geometa object of class `ISOMetadata` or `ISOFeatureCatalogue`

`metadataType` metadata type. By default `METADATA`

`uuidProcessing` UUID processing. By default `NOTHING`. Other possible value: `OVERWRITE`

`group` group

`category` category

`rejectIfInvalid` reject if invalid. Default `FALSE`

`publishToAll` publish to all. Default `TRUE`

`transformWith` transform with. Default is _none_

`schema` schema

`extra` extra

`geometa_validate` validate geometa object

`geometa_inspire` validate geometa object vs. INSPIRE

`geometa_inspireValidator` geometa INSPIRE validator to use

Method `updateMetadata()`**:** Inserts a metadata by file, XML object or **geometa** object of class `ISOMetadata` or `ISOFeatureCatalogue`. Extra parameters related to **geometa** objects: `geometa_validate` (TRUE by default) and `geometa_inspire` (FALSE by default) can be used to perform ISO and INSPIRE validation respectively. In that case on object of class `geometa::INSPIREMetadataValidator`, with a proper user API key, should be specified as `geometa_inspireValidator` argument.

Usage:

```
GNOpenAPIManager$updateMetadata(
  xml = NULL,
  file = NULL,
  geometa = NULL,
  metadataType = "METADATA",
  group,
  category = NULL,
  rejectIfInvalid = FALSE,
  publishToAll = TRUE,
  transformWith = "_none_",
  schema = NULL,
  extra = NULL,
  geometa_validate = TRUE,
  geometa_inspire = FALSE,
  geometa_inspireValidator = NULL
)
```

Arguments:

`xml` object of class XMLInternalNode-class from **XML**

`file` file

`geometa` geometa object of class `ISOMetadata` or `ISOFeatureCatalogue`

`metadataType` metadata type. By default `METADATA`

`group` group

`category` category

`rejectIfInvalid` reject if invalid. Default `FALSE`

`publishToAll` publish to all. Default `TRUE`

`transformWith` transform with. Default is `_none_`

`schema` schema

`extra` extra

`geometa_validate` validate geometa object

`geometa_inspire` validate geometa object vs. INSPIRE

`geometa_inspireValidator` geometa INSPIRE validator to use

Method `deleteMetadata()`: Deletes a metadata by ID

Usage:

```
GNOpenAPIManager$deleteMetadata(id, withBackup = TRUE)
```

Arguments:

`id` id

`withBackup` proceed with backup. Default is `TRUE`

Method `uploadAttachment()`: Uploads attachment

Usage:

```
GNOpenAPIManager$uploadAttachment(
  id,
  file,
  visibility = "public",
  approved = TRUE
)
```

Arguments:

`id` metadata identifier

`file` file to upload

`visibility` public or private

`approved` object of class `logical`

Returns: a named list of the uploaded attachment, including the url, size, id and type, `NULL` otherwise

Method `publishThumbnail()`: Publishes thumbnail based on URL

Usage:

```
GNOpenAPIManager$publishThumbnail(id, url, desc = "")
```

Arguments:

`id` metadata identifier

`url` thumbnail URL

`desc` thumbnail description

Returns: `TRUE` if published, `FALSE` otherwise

Method `doiCheckPreConditions()`: Checks pre-conditions to publish DOI

Usage:

```
GNOpenAPIManager$doiCheckPreConditions(id)
```

Arguments:

`id` metadata identifier

Returns: `TRUE` if DOI pre-conditions are fulfiled, `FALSE` otherwise

Method `createDOI()`**:** Submit a record to the Datacite metadata store in order to create a DOI.

Usage:

```
GNOpenAPIManager$createDOI(id)
```

Arguments:

`id` metadata identifier

Returns: `TRUE` if metadata record has been submitted with DOI created, `FALSE` otherwise

Method `deleteDOI()`**:** Remove a DOI (this is not recommended, DOI are supposed to be persistent once created. This is mainly here for testing).

Usage:

```
GNOpenAPIManager$deleteDOI(id)
```

Arguments:

`id`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNOpenAPIManager$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
## Not run:
  GNOpenAPIManager$new("http://localhost:8080/geonetwork", "admin", "geonetwork", "4.0.5")

## End(Not run)
```

GNPrivConfiguration *A GeoNetwork privilege configuration*

Description

This class is an utility to configure privileges

This class is an utility to configure privileges

Format

R6Class object.

R6Class object.

Details

GeoNetwork REST API - GeoNetwork privilege configuration

GeoNetwork REST API - GeoNetwork privilege configuration

Value

Object of R6Class for modelling a GeoNetwork Privilege configuration

Object of R6Class for modelling a GeoNetwork Privilege configuration

Public fields

`group` group

`privileges` privileges

Methods

Public methods:

- `GNPriv$new()`
- `GNPriv$clone()`

Method `new()`**:** Initializes a GNPriv object

Usage:

```
GNPriv$new(group, privileges)
```

Arguments:

`group` group

`privileges` privileges

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNPriv$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Public fields

`privileges` privileges

Methods

Public methods:

- `GNPrivConfiguration$new()`
- `GNPrivConfiguration$setPrivileges()`
- `GNPrivConfiguration$clone()`

Method `new()`**:** Initializes an object of class GNPrivConfiguration

Usage:

```
GNPrivConfiguration$new()
```

Method `setPrivileges()`**:** Sets the operation privileges for a particular group. Allowed group values are "guest","intranet" and "all". Allowed values for operation privileges are "view", "download", "editing", "notify", "dynamic" and "featured".

Usage:

```
GNPrivConfiguration$setPrivileges(group, privileges)
```

Arguments:

`group` group

`privileges` privileges

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNPrivConfiguration$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
## Not run:
 priv <- GNPriv$new(group="all", privileges=c("view","dynamic","featured"))

## End(Not run)

## Not run:
 pcfg <- GNPrivConfiguration$new()
 pcfg$setPrivileges("all", c("view","dynamic","featured"))

## End(Not run)
```

GNRESTRequest *GeoNetwork REST API REST Request*

Description

GeoNetwork REST API REST Request

GeoNetwork REST API REST Request

Format

R6Class object.

Value

Object of R6Class for modelling a GeoNetwork REST request

Public fields

`rootName` root name

`children` children

Methods

Public methods:

- GNRESTRequest$new()
- GNRESTRequest$setChild()
- GNRESTRequest$encode()
- GNRESTRequest$clone()

Method `new()`**:** Initializes a GNRESTRequest

Usage:

```
GNRESTRequest$new(...)
```

Arguments:

`...` any parameter to pass to the request

Method `setChild()`**:** Set child

Usage:

```
GNRESTRequest$setChild(key, value)
```

Arguments:

`key` key

`value` value

Method `encode()`**:** Encodes request as XML

Usage:

```
GNRESTRequest$encode()
```

Returns: an object of class `character` representing the XML

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNRESTRequest$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

`GNUtils` *GeoNetwork REST API Manager Utils*

Description

GeoNetwork REST API Manager Utils

GeoNetwork REST API Manager Utils

Format

`R6Class` object.

Value

Object of `R6Class` with static util methods for communication with the REST API of a GeoNetwork instance.

Static methods

`getUserAgent()` This method is used to get the user agent for performing GeoNetwork API requests. Here the user agent will be compound by geonapi package name and version.

`getUserToken(user, pwd)` This method is used to get the user authentication token for performing GeoNetwork API requests. Token is given a Base64 encoded string.

`GET(url, path, token, verbose)` This method performs a GET request for a given `path` to GeoNetwork REST API

`PUT(url, path, token, filename, contentType, verbose)` This method performs a PUT request for a given `path` to GeoNetwork REST API, to upload a file of name `filename` with given `contentType`

`POST(url, path, token, content, contentType, encode, verbose)` This method performs a POST request for a given `path` to GeoNetwork REST API, to post content of given `contentType`

`DELETE(url, path, token, verbose)` This method performs a DELETE request for a given GeoNetwork resource identified by a `path` in GeoNetwork REST API `parseResponseXML(req)` Convenience method to parse XML response from GeoNetwork REST API. Although package **httr** suggests the use of **xml2** package for handling XML, **geonapi** still relies on the package **XML**. Response from **httr** is retrieved as text, and then parsed as XML 'xmlParse' function.

`getPayloadXML(obj)` Convenience method to create payload XML to send to GeoNetwork.

Methods

Public methods:

- `GNUtils$clone()`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNUtils$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

`GNVersion` *A GeoNetwork version*

Description

This class is an utility wrap the Geonetwork version

Format

`R6Class` object.

Details

GeoNetwork REST API - GeoNetwork Version

Value

Object of `R6Class` for modelling a GeoNetwork version

Public fields

`version` version

`value` value

Methods

Public methods:

- `GNVersion$new()`
- `GNVersion$lowerThan()`
- `GNVersion$greaterThan()`
- `GNVersion$equalTo()`
- `GNVersion$clone()`

Method `new()`**:** Initializes an object of class GNVersion

Usage:

```
GNVersion$new(version)
```

Arguments:

`version` version

Method `lowerThan()`**:** Compares to a version and returns TRUE if it is lower, FALSE otherwise

Usage:

```
GNVersion$lowerThan(version)
```

Arguments:

`version` version

Returns: `TRUE` if lower, `FALSE` otherwise

Method `greaterThan()`**:** Compares to a version and returns TRUE if it is greater, FALSE otherwise

Usage:

```
GNVersion$greaterThan(version)
```

Arguments:

`version` version

Returns: `TRUE` if lower, `FALSE` otherwise

Method `equalTo()`**:** Compares to a version and returns TRUE if it is equal, FALSE otherwise

Usage:

```
GNVersion$equalTo(version)
```

Arguments:

`version` version

Returns: `TRUE` if lower, `FALSE` otherwise

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GNVersion$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Author(s)

Emmanuel Blondel <emmanuel.blondel1@gmail.com>

Examples

```
## Not run:
version <- GNVersion$new("2.6.4")

## End(Not run)
```

Index