Package 'Rodam'

March 21, 2019

Title Wrapper Functions for 'ODAM' (Open Data for Access and Mining) Web Services

Version 0.1.6

Date 2019-03-21

Copyright Institut National de la Recherche Agronomique (INRA)

URL https://github.com/INRA/ODAM

Maintainer Daniel Jacob <daniel.jacob@inra.fr>

Description 'ODAM' (Open Data for Access and Mining) is a framework that implements a simple way to make research data broadly accessible and fully available for reuse, including by a script language such as R. The main purpose is to make a data set accessible online with a minimal effort from the data provider, and to allow any scientists or bioinformaticians to be able to explore the data set and then extract a subpart or the totality of the data according to their needs. The Rodam package has only one class, 'odamws', that provides methods to allow you to retrieve online data using 'ODAM' Web Services. This obviously requires that data are implemented according the 'ODAM' approach , namely that the data subsets were deposited in the suitable data repository in the form of TSV files associated with their metadata also described in TSV files. See <http://www.slideshare.net/danieljacob771282/odam-open-data-access-and-mining>.

Depends R (>= 3.1.1), methods, RCurl (>= 1.95)

License LGPL (>= 3)

Encoding UTF-8

Suggests data.tree, knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.1.0

NeedsCompilation no

Author Daniel Jacob [cre, aut] (<https://orcid.org/0000-0002-6687-7169>)

Repository CRAN

Date/Publication 2019-03-21 14:13:24 UTC

R topics documented:

`Rodam-package` *Wrapper Functions for 'ODAM' (Open Data for Access and Mining) Web Services*

Description

'ODAM' (Open Data for Access and Mining) is a framework that implements a simple way to make research data broadly accessible and fully available for reuse, including by a script language such as R. The main purpose is to make a dataset accessible online with a minimal effort from the data provider, and to allow any scientists or bioinformaticians to be able to explore the dataset and then extract a subpart or the totality of the data according to their needs. To install the 'ODAM' software suite on your own server, this requires a machine (virtual or not) with a recent OS that support Docker. See [1].

The Rodam package has only one class, `odamws` that provides methods to allow you to retrieve online data using 'ODAM' Web Services. This obviously requires that data are implemented according the 'ODAM' approach , namely that the data subsets were deposited in the suitable data repository in the form of TSV files associated with their metadata also described in TSV files. See [2] for more details.

Author(s)

Maintainer: Daniel Jacob

References

1. Docker images of the ODAM software suite https://hub.docker.com/r/odam/getdata/
2. Presentation of the ODAM framework http://fr.slideshare.net/danieljacob771282/odam-open-data-access-and-mining
3. FIRM's data subsets of TSV files consistent with the ODAM framework https://doi.org/10.5281/zenodo.154041
4. Data Explorer online https://pmb-bordeaux.fr/dataexplorer/

See Also

The class that actually implements the API layer for the 'ODAM' web services `odamws`

odamws-class *API layer for the ODAM web services*

Description

the class that implements the API layer for the ODAM (Open Data for Access and Mining) web services.

Provides functions to allow you to retrieve online data using ODAM Web Services. This obviously requires that data are implemented according the ODAM approach (Open Data for Access and Mining), namely that the data subsets were deposited in the suitable data repository in the form of TSV files associated with their metadata also described in TSV files.

Fields

`wsURL` defines the URL of the webservice - Must be specify when creating a new instance of the odamws object.

`dsname` specifies the name of the Dataset to query - Must be specify when creating a new instance of the odamws object.

`delimiter` specifies the delimiter used within data subset files

`auth` specifies the authentication code to access to the Dataset by this webservice (if required)

`subsets` a data.frame object containing metadata related to the data subsets - Initialized during the instantiation step

`subsetNames` a list of the data subset names - Initialized during the instantiation step

`connectList` a matrix of the connection graph between data subsets (i.e. the links between each subset with the subset at its origin, so that links can be interpreted as 'obtained from'). The data subsets are referred by their subset number. (corresponding to the 'SetID' column in the 'subsets' field) - Initialized during the instantiation step.

Methods

`getCommonID(refID, setName1, setName2)` Returns the list of identifiers (defined by refID as an identifier attribute label) that are in common between two subsets (defined by the attribute label of the setName1 and setName2 subsets) i.e. resulting in the intersection of the two identifier sets.

`getDataByName(setName, condition = "")` Returns the data of the 'setName' subset as a data.frame

`getDataTree()` Returns a data.tree object filled up according to the data subset metadata

`getMerged(refID, setName1, setName2)` [DEPRECATED] Returns a data.frame containing data obtained by merging two subsets (defined by the attribute label of the setName1 and setName2 subsets) that have the same identifiers in common (defined by refID as an identifier attribute label) i.e. resulting in the intersection of the two identifier sets.

`getSetInCommon(setNameList)` Get the data subset in common with the data subset list 'setNameList'. Returns a list containing the elements :

* refID: Main Keyname serving as reference ID along with all data subsets defined in setNameList,

* setName : the data subset name corresponding to the refID

`getSubsetByName(setNameList, condition = "")` Returns both data and metadatas of the subsets defined by 'setNameList' as a list of objects. 'setNameList' can contain one or more subset names. If 'setNameList' contains two or more subset names, the returned data set will correspond to the merged data subsets based on the identifiers of the first common data subset :

data - a data.frame object containing the data. The column names of this data.frame are gathered according their categories and avaivalble in embedded lists, and described below.

varnames, facnames, qualnames, - Return lists containing the 'quantitative' variables, the 'factor' variables, the 'qualitative ' variables respectively.

varsBySubset - a list containing the 'quantitative' variables by subset.

idSet - a data.frame containing the metadata about the common identifier, namely the subset name it belongings (Subset), the identifier name (Attribute), the description (Description), the type (Type), and the CVTerm (CV_Term_ID, CV_Term_Name).

idName - the identifier name (Attribute) of the first common data subset.

LABELS - a data.frame containing the metadata about all attributes - its format is the same as the 'samplename' data.frame.

WSEntry - a data.frame containing the correspondance between some attributes and their alias name, these latter serving within a query to put a constraint a or selection on this attribute. Note: a 'WSEntry' is an alias name associated with an attribute that allows user to query the data subset by putting a filter condition (i.e. a selection constraint) on the corresponding attribute. Not all attributes have a WSEntry but only few ones, especially the attributes within the identifier and factor categories. For instance, the WSEntry of the 'SampleID' attribute is 'sample'. Thus, if you want to select only samples with their ID equal to 365, you have to specify the filter condition as 'sample/365'.

`getUpSetTable(setNameList)` Return an encoded dataframe in binary and set up so that columns represent data subsets present in 'setNameList', and each row represents an element (ID). If an element (ID) is in the data subset it is represented as a 1 in that position, otherwise it is represented as a 0. Useful for use with the R package UpSetR (https://cran.r-project.org/package=UpSetR)

`getWS(query = "")` Low level routine allowing to retrieve data or metadata from a query formatted according the ODAM framework specifications - Returns a data.frame object. By default, i.e. with an empty query, a data.frame object containing metadata related to the data subsets is returned.

Author(s)

Daniel Jacob - INRA UMR 1332 BFP (C) 2019

Examples

```
## Not run:
dh <- new("odamws", "https://pmb-bordeaux.fr/getdata/", "frim1")
dn <- show(dh)
# Get data from 'samples' subset with a constraint
data <- dh$getDataByName('samples','sample/365')
# Get 'activome' data subset
ds <- dh$getSubsetByName('activome')
# Get the merged data of both data subsets based on their common identifiers
setNameList <- c("activome", "qNMR_metabo" )
```

```
dsMerged <- dh$getSubsetByName(setNameList)

## End(Not run)
```

Index