Package ‘ExperimentHubData’

April 15, 2019

Type Package

Title Add resources to ExperimentHub

Version 1.8.0

biocViews Infrastructure, DataImport, GUI, ThirdPartyClient

Maintainer Bioconductor Package Maintainer <`maintainer@bioconductor.org`>

Description Functions to add metadata to ExperimentHub db and resource files to AWS S3 buckets.

License Artistic-2.0

Depends utils, BiocGenerics (>= 0.15.10), S4Vectors, AnnotationHubData (>= 1.11.6)

Imports methods, ExperimentHub, BiocManager, DBI, BiocCheck, httr, curl, biocViews, graph

Suggests GenomeInfoDb, RUnit, knitr, BiocStyle

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/ExperimentHubData

git_branch RELEASE_3_8

git_last_commit 2bc4d99

git_last_commit_date 2018-10-30

Date/Publication 2019-04-15

Author Bioconductor Maintainer [cre]

R topics documented:

`addResources` *addResources*

Description

Add resource metadata to a local ExperimentHub database

Usage

```
addResources(pathToPackage, fileName=character(), insert = FALSE, ...)
```

Arguments

`pathToPackage`	Full path to data package including package name.
`fileName`	Name of single metadata file located in "inst/extdata". If none is provided the function looks for a file named "metadata.csv".
`insert`	A `logical` to control if metadata are inserted in the ExperimentHub db. By default this option is FALSE which is a useful state in which to test a new recipe and confirm the metadata fields are correct. When `insert = TRUE`, the "EXPERIMENT_HUB_SERVER_POST_URL" global option must be set to the http location of the ExperimentHubServer in the global environment or .Rprofile. This option controls Additionally, AWS command line tools must be installed on the local machine to push files to S3 buckets. See https://aws.amazon.com/cli/ for installation instructions.
`...`	TDB. Currently not used.

Details

This function is used by the Bioconductor Core team to add new metadata to the production database.

When `insert` is TRUE, `addResources` attempts to add the metadata to the local database. (NOTE: A local database can be created with the ExperimentHub docker). Records in ExperimentHub must have unique file names. If the new metadata have duplicate file names a warning is thrown and the records are omitted from those added to the database.

This function does not add data to an AWS S3 bucket. ExperimentHub packages do not have 'recipes' that generate data on the fly. Instead, data files are provided by the maintainer in final form and added to the appropriate S3 location in a separate step.

Value

A list of ExperimentHubMetadata objects.

See Also

- ExperimentHubMetadata
- AnnotationHubMetadata
- makeExperimentHubMetadata

Examples

```
    ## Not run: 
      ## Generate metadata for inspection
      addResources("/home/vobencha/mypackage", insert=FALSE)
      ## Inset metadata into ExperimentHub database
      addResources("/home/vobencha/mypackage", insert=TRUE)
    
  ## End(Not run)
```

`ExperimentHubMetadata-class`

Class ExperimentHubMetadata objects and methods

Description

The ExperimentHubMetadata object is used to represent records in the server data base.

Usage

```
ExperimentHubMetadata(ExperimentHubRoot=NA_character_,
        BiocVersion=BiocManager::version(),
        SourceUrl=NA_character_,
        SourceType=NA_character_,
        SourceVersion=NA_character_,
        SourceLastModifiedDate=as.POSIXct(NA_character_),
        SourceMd5=NA_character_,
        SourceSize=NA_real_,
        DataProvider=NA_character_,
        Title=NA_character_,
        Description=NA_character_,
        Maintainer=NA_character_,
        Species=NA_character_,
        TaxonomyId=NA_integer_,
        Genome=NA_character_,
        Tags=NA_character_,
        RDataClass=NA_character_,
        RDataDateAdded=as.POSIXct(NA_character_),
        RDataPath=NA_character_,
        Coordinate_1_based=TRUE,
        Notes=NA_character_,
        DispatchClass=NA_character_,
        PreparerClass=NA_character_,
        Location_Prefix='http://s3.amazonaws.com/experimenthub/')
```

Arguments

`ExperimentHubRoot`	`character(1)` Prefix of the local path to resources to be added to ExperimentHub. Internal use only.
`SourceUrl`	`character()` URL of original resource(s).
`SourceType`	`character()` Form of original data, e.g., BED, FASTA, etc.

`SourceVersion`	`character(1)` Version of original file.
`SourceLastModifiedDate`	`POSIXct()` Date when resource was last modified.
`SourceMd5`	`character(1)` md5 hash of original file.
`SourceSize`	`numeric(1)` Number of bytes in original file.
`DataProvider`	`character(1)` Provider of original data, e.g., NCBI, UniProt etc.
`Title`	`character(1)` Title for the resource with version or genome build as appropriate. Titles must be unique and not match any existing title in ExperimentHub.
`Description`	`character(1)` Description of the resource. May include details such as data type, format, study origin, sequencing technology, treated vs control, number of samples etc.
`Species`	`character(1)` Species name.
`TaxonomyId`	`character(1)` NCBI code.
`Genome`	`character(1)` Name of genome build.
`Tags`	'Tags' are search terms used to define a subset of resources in a `Hub` object, e.g, in a call to `query`. For ExperimentHub resources, 'Tags' are automatically generated from the 'biocViews' in the DESCRIPTION file of the accompanying software package. 'Tags' values supplied by the user are not be entered in the database and are not part of the formal metadata. This 'controlled vocabulary' approach was taken to limit the search terms to a well defined set and may change in the future.
`RDataClass`	`character(1)` Class of derived object (e.g. 'GRanges').
`RDataDateAdded`	`POSIXct()` Date resource was added to ExperimentHub. The default is today's date and is auto-generated when metadata are constructed. Resources will appear in snapshots with a date greater than or equal to the `RDataDateAdded`.
`RDataPath`	`character(1)` File path to where object is stored in AWS S3 bucket or on the web.
`Maintainer`	`character(1)` Maintainer name and email address, 'A Maintainer `a.maintainer@email.addr`'
`BiocVersion`	`character(1)`. The first Bioconductor version the resource was made available for. Unless removed from the hub, the resource will be available for all versions greater than or equal to this field.
`Coordinate_1_based`	`logical(1)` Do coordinates start with 1 or 0?
`DispatchClass`	`character(1)`. Determines how data are loaded into R. The value for this field should be 'Rda' if the data were serialized with `save()` and 'Rds' if serialized with `saveRDS`. The filename should have the appropriate 'rda' or 'rds' extension. A number of dispatch classes are pre-defined in AnnotationHub/R/AnnotationHubResource-class.R with the suffix 'Resource'. For example, if you have sqlite files, the AnnotationHubResource-class.R defines SQLiteFileResource so the DispatchClass would be SQLiteFile. Contact maintainer@bioconductor.org if you are not sure which class to use.
`Location_Prefix`	`character(1)` URL location of AWS S3 bucket or web site where resource is located.
`Notes`	`character()` Notes about the resource.
`PreparerClass`	`character(1)` Used internally.

Details

In practice, instances of this class are generated by a call to `addResources` or `makeExperimentHubMetadata` instead of a direct call to the constructor.

`addResources` is a function used by the Bioconductor Core team when adding new metadata records to the production database. `makeExperimentHubMetadata` and the low-level helper

Value

A ExperimentHubMetadata object.

See Also

- addResources
- makeExperimentHubMetadata

Examples

```
showClass("ExperimentHubMetadata")
```

`makeExperimentHubMetadata` *Make ExperimentHubMetadata objects from csv file of metadata*

Description

Make ExperimentHubMetadata objects from metadata.csv file located in the "inst/extdata/" package directory of an ExperimentHub package.

Usage

```
makeExperimentHubMetadata(pathToPackage, fileName=character())
```

Arguments

`pathToPackage`	Full path to data package including the package name; no trailing slash
`fileName`	Name of single metadata file located in "inst/extdata". If none is provided the function looks for a file named "metadata.csv".

Details

- makeExperimentHubMetadata: Reads the resource metadata in the metadata.csv file into a ExperimentHubMetadata object. The ExperimentHubMetadata is inserted in the ExperimentHub database. Intended for internal use or package authors checking the validity of package metadata.

- Formatting metadata files:

  `makeExperimentHubMetadata` reads .csv files of metadata located in "inst/extdata". Internal functions perform checks for required columns and data types and can be used by package authors to validate their metadata before submitting the package for review.

  The rows of the .csv file(s) represent individual `Hub` resources (i.e., data objects) and the columns are the metadata fields. All fields should be a single character string of length 1.

  Required Fields in metadata file:

- Title: `character(1)`. Name of the resource. This can be the exact file name (if self-describing) or a more complete description.
- Description: `character(1)`. Brief description of the resource, similar to the 'Description' field in a package DESCRIPTION file.
- BiocVersion: `character(1)`. The first Bioconductor version the resource was made available for. Unless removed from the hub, the resource will be available for all versions greater than or equal to this field.
- Genome: `character(1)`. Genome.
- SourceType: `character(1)`. Format of original data, e.g., FASTA, BAM, BigWig, etc.
- SourceUrl: `character(1)`. Optional location of original data files. Multiple urls should be provided as a comma separated string.
- SourceVersion: `character(1)`. Version of original data.
- Species: `character(1)`. Species.
- TaxonomyId: `character(1)`. Taxonomy ID.
- Coordinate_1_based: `logical`. TRUE if data are 1-based.
- DataProvider: `character(1)`. Name of company or institution that supplied the original (raw) data.
- Maintainer: `character(1)`. Maintainer name and email in the following format: Maintainer Name <username@address>.
- RDataClass: `character(1)`. R / Bioconductor class the data are stored in, e.g., GRanges, SummarizedExperiment, ExpressionSet etc.
- DispatchClass: `character(1)`. Determines how data are loaded into R. The value for this field should be 'Rda' if the data were serialized with `save()` and 'Rds' if serialized with `saveRDS`. The filename should have the appropriate 'rda' or 'rds' extension.

  A number of dispatch classes are pre-defined in AnnotationHub/R/AnnotationHubResource-class.R with the suffix 'Resource'. For example, if you have sqlite files, the AnnotationHubResource-class.R defines SQLiteFileResource so the DispatchClass would be SQLiteFile. Contact maintainer@bioconductor.org if you are not sure which class to use.
- Location_Prefix: `character(1)`. Do not include this field if data are stored in the Bioconductor AWS S3; it will be generated automatically.

  If data will be accessed from a location other than AWS S3 this field should be the base url.
- RDataPath: `character(1)`.This field should be the remainder of the path to the resource. The `Location_Prefix` will be prepended to `RDataPath` for the full path to the resource. If the resource is stored in Bioconductor's AWS S3 buckets, it should start with the name of the package associated with the metadata and should not start with a leading slash. It should include the resource file name.
- Tags: `character() vector`. 'Tags' are search terms used to define a subset of resources in a `Hub` object, e.g, in a call to `query`.

  For ExperimentHub resources, 'Tags' are automatically generated from the 'biocViews' in the DESCRIPTION. 'Tags' values supplied by the user are not be entered in the database and are not part of the formal metadata. This 'controlled vocabulary' approach was taken to limit the search terms to a well defined set and may change in the future.

  'Tags' for AnnotationHub resources are a free-form field of search terms defined by the user. The package name is added as one of the 'Tags' before the metadata are finalized. Multiple 'Tags' are specified as a colon separated string, e.g., tags for two resources would look like this:

  ```
  Tags=c("tag1:tag2:tag3", "tag1:tag3")
  ```

NOTE: The metadata file can have additional columns beyond the 'Required Fields' listed above. These values are not added to the Hub database but they can be used in package functions to provide an additional level of metadata on the resources.

Value

A list of `ExperimentHubMetadata` objects.

See Also

- `addResources`
- `ExperimentHubMetadata` class
- makeAnnotationHubMetadata

Examples

```
## makeExperimentHubMetadata() reads data from inst/scripts/<files>.csv
## into ExperimentHubMetadata objects. These objects are used to insert
## metadata into the production database. This function is used internally
## by addResources() and is not intended to be called directly.

## For an example of how this works we can use the GSE62944 ExperimentHub
## package. Download the source tarball from:

# http://www.bioconductor.org/packages/devel/data/experiment/html/GSE62944.html

## and unpack it. Set 'pathToPackage' to point to the downloaded source.
## Then call the function:
## Not run:
makeExperimentHubMetadata("path/to/mypackage")

## End(Not run)
```

Index