Package 'ExperimentHub'

April 15, 2020

Type Package

Title Client to access ExperimentHub resources

Version 1.12.0

Author Bioconductor Package Maintainer <maintainer@bioconductor.org>

Maintainer Bioconductor Package Maintainer <maintainer@bioconductor.org>

Description This package provides a client for the Bioconductor ExperimentHub web resource. ExperimentHub provides a central location where curated data from experiments, publications or training courses can be accessed. Each resource has associated metadata, tags and date of modification. The client creates and manages a local cache of files retrieved enabling quick and reproducible access.

License Artistic-2.0

biocViews Infrastructure, DataImport, GUI, ThirdPartyClient

Depends methods, BiocGenerics (>= 0.15.10), AnnotationHub (>= 2.17.9), BiocFileCache (>= 1.5.1)

Imports utils, S4Vectors, BiocManager, curl, rappdirs

Suggests knitr, BiocStyle

Enhances ExperimentHubData

VignetteBuilder knitr

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

git_branch RELEASE_3_10

git_last_commit f0406a8

git_last_commit_date 2019-10-29

Date/Publication 2020-04-14

R topics documented:

ExperimentHub-objects *ExperimentHub objects and their related methods and functions*

Description

Use ExperimentHub to interact with Bioconductor's ExperimentHub service. Query the instance to discover and use resources that are of interest, and then easily download and import the resource into R for immediate use.

Use ExperimentHub() to retrieve information about all records in the hub. If working offline, add argument localHub=TRUE to work with a local, non-updated hub; It will only have resources available that have previously been downloaded. To force redownload of the hub, refreshHub(hubClass="ExperimentHub") can be utilized.

Discover records in a hub using mcols(), query(), subset(), [, and display().

Retrieve individual records using [[. On first use of a resource, the corresponding files or other hub resources are downloaded from the internet to a local cache. On this and all subsequent uses the files are quickly input from the cache into the R session. If a user wants to download the file again and not use the cache version add the argument force=TRUE.

ExperimentHub records can be added (and sometimes removed) at any time. snapshotDate() restricts hub records to those available at the time of the snapshot. possibleDates() lists snapshot dates valid for the current version of Bioconductor.

The location of the local cache can be found (and updated) with getExperimentHubOption and setExperimentHubOption; removeCache removes all cache resources.

Constructors

ExperimentHub(...,hub=getExperimentHubOption("URL"),cache=getExperimentHubOption("CACHE"),proxy

Create an ExperimentHub instance, possibly updating the current database of records.

Accessors

In the code snippets below, x and object are ExperimentHub objects.

hubCache(x): Gets the file system location of the local ExperimentHub cache.

hubUrl(x): Gets the URL for the online hub.

length(x): Get the number of hub records.

names(x): Get the names (ExperimentHub unique identifiers, of the form AH12345) of the hub records.

fileName(x): Get the file path of the hub records as stored in the local cache (ExperimentHub files are stored as unique numbers, of the form 12345). NA is returned for those records which have not been cached.

package(x): Returns a named character vector of package name associated with the hub resource.

recordStatus(x,record): Returns a data.frame of the record id and status. x must be a Hub object and record must be a character(1). Can be used to discover why a resource was removed from the hub.

mcols(x): Get the metadata columns describing each record. Columns include:

title Record title, frequently the file name of the object.

dataprovider Original provider of the resource, e.g., Ensembl, UCSC.

species The species for which the record is most relevant, e.g., 'Homo sapiens'.

taxonomyid NCBI taxonomy identifier of the species.

genome Genome build relevant to the record, e.g., hg19.

description Textual description of the resource, frequently automatically generated from file path and other information available when the record was created.

tags Single words added to the record to facilitate identification, e.g,. TCGA, Roadmap.

rdataclass The class of the R object used to represent the object when imported into R, e.g., `GRanges`, `VCFFile`.

sourceurl Original URL of the resource.

sourectype Format of the original resource, e.g., BED file.

Subsetting and related operations

In the code snippets below, `x` is an ExperimentHub object.

`x$name`: Convenient reference to individual metadata columns, e.g., `x$species`.

`x[i]`: Numerical, logical, or character vector (of ExperimentHub names) to subset the hub, e.g., `x[x$species == "Homo sapiens"]`.

`x[[i,force=FALSE,verbose=TRUE]]`: Numerical or character scalar to retrieve (if necessary) and import the resource into R. If a user wants to download the file again and not use the cache version add the argument `force=TRUE`. `verbose=FALSE` will quiet status messages.

`query(x,pattern,ignore.case=TRUE,pattern.op=`\`&\`): Return an ExperimentHub subset containing only those elements whose metadata matches `pattern`. Matching uses `pattern` as in `grepl` to search the `as.character` representation of each column, performing a logical `&` across columns. e.g., `query(x,c("Homo sapiens","hg19","GTF"))`.

- `pattern` A character vector of patterns to search (via `grepl`) for in any of the `mcols()` columns.
- `ignore.case` A logical(1) vector indicating whether the search should ignore case (TRUE) or not (FALSE).
- `pattern.op` Any function of two arguments, describing how matches across pattern elements are to be combined. The default `&` requires that only records with *all* elements of `pattern` in their metadata columns are returned. `&`, `|` and `!` are most notably available. See `"?&"` or `?base::Ops` for more information.

`subset(x,subset)`: Return the subset of records containing only those elements whose metadata satisfies the *expression* in `subset`. The expression can reference columns of `mcols(x)`, and should return a logical vector of length `length(x)`. e.g., `subset(x,species == "Homo sapiens" & genome=="GRCh38")`.

`display(object)`: Open a web browser allowing for easy selection of hub records via interactive tabular display. Return value is the subset of hub records identified while navigating the display.

`listResources(hub,package,filterBy=character())`: List resources in ExperimentHub associated with `package`. `filterBy` is a character vector of search terms.

`loadResources(hub,package,filterBy=character())`: Load resources in ExperimentHub associated with `package`. `filterBy` is a character vector of search terms.

Cache and hub management

In the code snippets below, `x` is an ExperimentHub object.

`snapshotDate(x)` and `snapshotDate(x) <-value`: Gets or sets the date for the snapshot in use. `value` should be one of `possibleDates()`.

`possibleDates(x)`: Lists the valid snapshot dates for the version of Bioconductor that is being run (e.g., BiocManager::version()).

`cache(x)` and `cache(x) <-NULL`: Adds (downloads) all resources in `x`, or removes all local resources corresponding to the records in `x` from the cache. In this case, `x` would typically be a small subset of ExperimentHub resources.

`hubUrl(x)`: Gets the URL for the online ExperimentHub.

`hubCache(x)`: Gets the file system location of the local ExperimentHub cache.

`refreshHub(...,hub,cache,proxy,hubClass=c("AnnotationHub","ExperimentHub"))`: Force redownload of Hub sqlite file. This returns a Hub object as if calling the constructor (ie. ExperimentHub()). For force redownload specifically for ExperimentHub the base call should be `refreshHub(hubClass="ExperimentHub")`

`removeCache(x)`: Removes local ExperimentHub database and all related resources. After calling this function, the user will have to download any ExperimentHub resources again.

Coercion

In the code snippets below, `x` is an ExperimentHub object.

`as.list(x)`: Coerce x to a list of hub instances, one entry per element. Primarily for internal use.

`c(x,...)`: Concatenate one or more sub-hub. Sub-hubs must reference the same ExperimentHub instance. Duplicate entries are removed.

For developers

`createHubAccessors(pkgname,titles)`: This helper is intended to be used by ExperimentHub package developers in their .onLoad() function in zzz.R. It generates and exports functions by resource name which allows the resource to be loaded with resource123() in addition to the standard method via the ExperimentHub interface, e.g., ehub[["EH123"]]. When 'metadata=TRUE' just the metadata are displayed, when 'metadata=FALSE' the full resource is loaded, e.g, resource123(metadata=TRUE). See vignette for more information.

Author(s)

Bioconductor Core Team

Examples

```
## Create an ExperimentHub object
ehub <- ExperimentHub()
ehub

## Display packages associated with resources
unique(package(ehub))
query(ehub, "alpineData")

## Search metadata by general terms
query(ehub, c("FASTQ", "Homo sapiens"))
```

`getExperimentHubOption` *Get and set options for default ExperimentHub behavior.*

Description

These functions get or set options for creation of new 'ExperimentHub' instances.

Usage

```
getExperimentHubOption(arg)
setExperimentHubOption(arg, value)
```

Arguments

`arg`	The character(1) hub options to set. see 'Details' for current options.
`value`	The value to be assigned to the hub option.

Details

Supported options include:

"URL": character(1). The base URL of the Experiment Hub. Default: https://experimenthub.bioconductor.org

"CACHE": character(1). The location of the hub cache. Default: ".ExperimentHub" in the user home directory.

"MAX_DOWNLOADS": numeric(1). The integer number of downloads allowed before triggering an error. This is to help avoid accidental download of a large number of ExperimentHub members.

"PROXY": `request` object returned by `httr::use_proxy()`. The `request` object describes a proxy connection allowing Internet access, usually through a restrictive firewall. Setting this option sends all ExperimentHub requests through the proxy. Default: NULL.

In `setHubOption("PROXY",value)`, `value` can be one of NULL, a `request` object returned by `httr::use_proxy()`, or a well-formed URL as character(1). The URL can be completely specified by `http://username:password@proxy.dom.com:8080`; `username:password` and port (e.g. `:8080`) are optional.

"LOCAL": logical(1). TRUE/FALSE should the ExperimentHub create a hub consisting only of previously downloaded resourcesd. Default: FALSE.

Default values may also be determined by system and global R environment variables visible *before* the package is loaded. Use options or variables preceded by "EXPERIMENT_HUB_", e.g., `options(EXPERIMENT_HUB_MAX_DOWNLOADS=10)` prior to package load sets the default number of downloads to 10.

Value

The requested or successfully set option.

Author(s)

Bioconductor Core Team

Examples

```
getExperimentHubOption("URL")
## Not run:
setExperimentHubOption("CACHE", "~/.myHub")

## End(Not run)
```

---

utilities *Utility functions for discovering package-specific Hub resources.*

Description

List and load resources from ExperimentHub filtered by package name and optional search terms.

Usage

```
listResources(hub, package, filterBy = character())
loadResources(hub, package, filterBy = character())
```

Arguments

hub	A Hub object; currently only meaningful for ExperimentHub.
package	A character(1) name of a package with resources hosted in the Hub.
filterBy	A character() vector of search terms for additional filtering. Can be any terms found in the metadata (mcols()) of the resources. When not provided, there is no additional filtering and all resources associated with the given package are returned.

Details

Currently listResources and loadResources are only meaningful for ExperimentHub objects.

Value

listResources returns a character vector; loadResources returns a list of data objects.

Examples

```
## Not run:
## Packages with resources hosted in ExperimentHub:
require(ExperimentHub)
eh <- ExperimentHub()
unique(package(eh))

## All resources associated with the 'GSE62944' package:
listResources(eh, "GSE62944")

## Resources associated with the 'curatedMetagenomicData' package
## filtered by 'plaque.abundance':
listResources(eh, "curatedMetagenomicData", "plaque.abundance")

## 'loadResources()' returns a list of the data objects:
```

```
loadResources(eh, "curatedMetagenomicData", "plaque.abundance")

## End(Not run)
```

Index