Package ‘cBioPortalData’

June 22, 2025

Title Exposes and Makes Available Data from the cBioPortal Web Resources

Version 2.20.0

Description The cBioPortalData R package accesses study datasets from the cBio Cancer Genomics Portal. It accesses the data either from the pre-packaged zip / tar files or from the API interface that was recently implemented by the cBioPortal Data Team. The package can provide data in either tabular format or with MultiAssayExperiment object that uses familiar Bioconductor data representations.

Depends R (>= 4.5.0), AnVIL (>= 1.19.5), MultiAssayExperiment

Imports BiocBaseUtils, BiocFileCache (>= 1.5.3), digest, dplyr, GenomeInfoDb, GenomicRanges, httr, IRanges, methods, readr, RaggedExperiment, RTCGAToolbox (>= 2.19.7), S4Vectors, SummarizedExperiment, stats, tibble, tidyr, TCGAutils (>= 1.9.4), utils

Suggests BiocStyle, jsonlite, knitr, survival, survminer, rmarkdown, testthat

Date 2025-04-07

License AGPL-3

Encoding UTF-8

VignetteBuilder knitr

URL `https://github.com/waldronlab/cBioPortalData`

BugReports `https://github.com/waldronlab/cBioPortalData/issues`

biocViews Software, Infrastructure, ThirdPartyClient

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Collate 'utils.R' 'cBioDataPack.R' 'cBioPortal-class.R' 'cBioPortal.R' 'cBioPortalData-pkg.R' 'cBioPortalData.R' 'cache.R'

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

git_branch RELEASE_3_21

git_last_commit e2964c0

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-06-22

Author Levi Waldron [aut],
Marcel Ramos [aut, cre] (ORCID:
<https://orcid.org/0000-0002-3242-0582>),
Karim Mezhoud [ctb]

Maintainer Marcel Ramos <`marcel.ramos@sph.cuny.edu`>

Contents

`cBioCache` *Manage cache / download directories for study data*

Description

Managing data downloads is important to save disk space and avoid re-downloading data files. This can be done via the integrated `BiocFileCache` system.

Usage

```
cBioCache(..., ask = interactive())

setCache(
  directory = tools::R_user_dir("cBioPortalData", "cache"),
  verbose = TRUE,
  ask = interactive()
)

removePackCache(cancer_study_id, dry.run = TRUE)
```

Arguments

`...`	For `cBioCache`, arguments passed to `setCache`
`ask`	logical (default TRUE when interactive session) Confirm the file location of the cache directory
`directory`	The file location where the cache is located. Once set future downloads will go to this folder.
`verbose`	Whether to print descriptive messages
`cancer_study_id`	`character(1)` The `studyId` from `getStudies`
`dry.run`	logical Whether or not to remove cache files (default TRUE).

Value

cBioCache: The path to the cache location

cBioCache

Get the directory location of the cache. It will prompt the user to create a cache if not already created. A specific directory can be used via `setCache`.

setCache

Specify the directory location of the data cache. By default, it will go to the user directory as given by:

```
tools::R_user_dir("cBioPortalData", "cache")
```

removePackCache

Some files may become corrupt when downloading, this function allows the user to delete the tarball associated with a `cancer_study_id` in the cache. This only works for the `cBioDataPack` function. To remove the entire `cBioPortalData` cache, run `unlink("~/.cache/cBioPortalData")`.

Examples

```
cBioCache()

removePackCache("acc_tcga", dry.run = TRUE)
```

`cBioCache-deprecated` *Deprecated cache helper functions*

Description

`cBioPortalData` no longer caches data from API responses; therefore, `removeDataCache` is no longer needed. It will be removed as soon as the next release of Bioconductor.

Usage

```
removeDataCache(
  api,
  studyId = NA_character_,
  genePanelId = NA_character_,
  genes = NA_character_,
  molecularProfileIds = NULL,
  sampleListId = NULL,
  sampleIds = NULL,
  by = c("entrezGeneId", "hugoGeneSymbol"),
  dry.run = TRUE,
  ...
)
```

Arguments

`api`	An API object of class `cBioPortal` from the `cBioPortal` function
`studyId`	`character(1)` Indicates the "studyId" as taken from `getStudies`
`genePanelId`	`character(1)` Identifies the gene panel, as obtained from the `genePanels` function
`genes`	`character()` Either Entrez gene identifiers or Hugo gene symbols. When included, the 'by' argument indicates the type of identifier provided and 'genePanelId' is ignored. Preference is given to Entrez IDs due to faster query responses.
`molecularProfileIds`	`character()` A vector of molecular profile IDs
`sampleListId`	`character(1)` A sample list identifier as obtained from `sampleLists()`
`sampleIds`	`character()` Sample identifiers
`by`	`character(1)` Either 'entrezGeneId' or 'hugoGeneSymbol' for row metadata (default: 'entrezGeneId')
`dry.run`	logical Whether or not to remove cache files (default TRUE).
`...`	Additional arguments to lower level API functions

Value

removeDataCache: The path to the cache location when `dry.run = FALSE` if the file exists. Otherwise, when `dry.run = TRUE`, the function return the output of the `file.remove` operation.

removeDataCache

Remove the computed cache location based on the function inputs to `cBioPortalData()`. To remove the cache, simply replace the `cBiocPortalData()` function name with `removeDataCache()`; see the example. If the computed cache location is not found, it will return an empty vector.

Examples

```
cbio <- cBioPortal()

cBioPortalData(
    cbio, by = "hugoGeneSymbol",
    studyId = "acc_tcga",
    genePanelId = "AmpliSeq",
    molecularProfileIds =
        c("acc_tcga_rppa", "acc_tcga_linear_CNA", "acc_tcga_mutations")
)

removeDataCache(
    cbio, by = "hugoGeneSymbol",
    studyId = "acc_tcga",
    genePanelId = "AmpliSeq",
    molecularProfileIds =
        c("acc_tcga_rppa", "acc_tcga_linear_CNA", "acc_tcga_mutations"),
    dry.run = TRUE
)
```

`cBioDataPack` *Obtain pre-packaged data from cBioPortal and represent as a MultiAssayExperiment object*

Description

The `cBioDataPack` function allows the user to download and process cancer study datasets found in MSKCC's cBioPortal. Output datasets use the MultiAssayExperiment data representation to faciliate analysis and data management operations.

Usage

```
cBioDataPack(
  cancer_study_id,
  use_cache = TRUE,
  names.field = c("Hugo_Symbol", "Entrez_Gene_Id", "Gene"),
  cleanup = TRUE,
  ask = interactive(),
  check_build = TRUE
)
```

Arguments

`cancer_study_id`	`character(1)` The study identifier from cBioPortal as seen in the dataset links at `https://www.cbioportal.org/datasets`.
`use_cache`	`logical(1)` (default TRUE) create the default cache location and use it to track downloaded data. If data found in the cache, data will not be re-downloaded. A path can also be provided to data cache location.
`names.field`	`character()` Possible column names for the column that will used to label ranges for data such as mutations or copy number (defaults: "Hugo_Symbol", "Entrez_Gene_Id", "Gene", and "Composite.Element.REF"). Values are cycled through and eliminated when no data present, or duplicates are found. Values in the corresponding column must be unique in each row.
`cleanup`	`logical(1)` whether to delete the `untar`-red contents from the `exdir` folder (default TRUE)
`ask`	`logical(1)` Whether to prompt the the user before downloading and loading study `MultiAssayExperiment` that is not currently building based on previous testing. Set to `interactive()` by default. In a non-interactive session, data download will be attempted; equivalent to `ask = FALSE`. The argument will also be used when a cache directory needs to be created when using `downloadStudy`.
`check_build`	logical(1L) Whether to check the build status of the `studyId` using an internal dataset. This argument should be set to `FALSE` if using alternative `hostnames`, e.g., 'pedcbioportal.kidsfirstdrc.org'

Details

The full list of study identifiers (`studyIds`) can obtained from `getStudies()`. Currently, only ~ 72% of datasets can be represented as `MultiAssayExperiment` data objects from the data tarballs. Refer to `getStudies(..., buildReport = TRUE)` and its `"pack_build"` column to see which study identifiers are not building. Users who would like to prioritize particular datasets should open GitHub issues at the URL in the `DESCRIPTION` file. For a more fine-grained approach to downloading data from the cBioPortal API, refer to the `cBioPortalData` function.

Value

A MultiAssayExperiment object

cBio_URL

The `cBioDataPack` function accesses data from the `cBio_URL` option. By default, it points to an Amazon S3 bucket location. Previously, it pointed to 'http://download.cbioportal.org'. This recent change (> 2.1.17) should provide faster and more reliable downloads for all users. See the URL using `cBioPortalData:::.url_location`. This can be changed if there are mirrors that host this data by setting the `cBio_URL` option with `getOption("cBio_URL", "https://some.url.com/")` before running the function.

Author(s)

Levi Waldron, Marcel R., Ino dB.

See Also

https://www.cbioportal.org/datasets, cBioPortalData, removePackCache

Examples

```
cbio <- cBioPortal()

head(getStudies(cbio)[["studyId"]])

mae <- cBioDataPack("acc_tcga")
```

`cBioPortal` *The R interface to the cBioPortal API Data Service*

Description

This section of the documentation lists the functions that allow users to access the cBioPortal API. The main representation of the API can be obtained from the `cBioPortal` function. The supporting functions listed here give access to specific parts of the API and allow the user to explore the API with individual calls. Many of the functions here are listed for documentation purposes and are recommended for advanced usage only. Users should only need to use the `cBioPortalData` main function to obtain data.

Usage

```
cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api. = "/api/v2/api-docs",
  token = character()
)

getStudies(api, buildReport = FALSE)

clinicalData(api, studyId = NA_character_)

molecularProfiles(
  api,
  studyId = NA_character_,
  projection = c("SUMMARY", "ID", "DETAILED", "META")
)

fetchData(
  api,
  studyId,
  molecularProfileIds = NA_character_,
```

```
  entrezGeneIds = NULL,
  sampleIds = NULL
)

mutationData(
  api,
  molecularProfileIds = NA_character_,
  entrezGeneIds = NULL,
  sampleIds = NULL
)

molecularData(
  api,
  molecularProfileIds = NA_character_,
  entrezGeneIds = NULL,
  sampleIds = NULL
)

copyNumberData(
  api,
  molecularProfileIds = NA_character_,
  entrezGeneIds = NULL,
  sampleIds = NULL,
  sampleListId = NULL,
 discreteCopyNumberEventType = c("HOMDEL_AND_AMP", "HOMDEL", "AMP", "GAIN", "HETLOSS",
    "DIPLOID", "ALL"),
  projection = c("SUMMARY", "ID", "DETAILED", "META")
)

searchOps(api, keyword)

samplesInSampleLists(api, sampleListIds = NA_character_)

sampleLists(api, studyId = NA_character_)

allSamples(api, studyId = NA_character_)

getSampleInfo(
  api,
  studyId = NA_character_,
  sampleListIds = NULL,
  projection = c("SUMMARY", "ID", "DETAILED", "META")
)

genePanels(api)

getGenePanel(api, genePanelId = NA_character_)
```

```
genePanelMolecular(
  api,
  molecularProfileId = NA_character_,
  sampleListId = NULL,
  sampleIds = NULL
)

getGenePanelMolecular(api, molecularProfileIds = NA_character_, sampleIds)

geneTable(api, pageSize = 1000, pageNumber = 0, ...)

queryGeneTable(
  api,
  by = c("entrezGeneId", "hugoGeneSymbol"),
  genes = NA_character_,
  genePanelId = NA_character_
)

getDataByGenes(
  api,
  studyId = NA_character_,
  genes = NA_character_,
  genePanelId = NA_character_,
  by = c("entrezGeneId", "hugoGeneSymbol"),
  molecularProfileIds = NULL,
  sampleListId = NULL,
  sampleIds = NULL,
  ...
)
```

Arguments

`hostname`	`character(1)` The internet location of the service (default: 'www.cbioportal.org')
`protocol`	`character(1)` The internet protocol used to access the hostname (default: 'https')
`api.`	`character(1)` The directory location of the API protocol within the hostname (default: '/api/v2/api-docs')
`token`	`character(1)` The Authorization Bearer token e.g., "63eba81c-2591-4e15-9d1c-fb6e8e51e35d" or a path to text file.
`api`	An API object of class `cBioPortal` from the `cBioPortal` function
`buildReport`	`logical(1)` Indicates whether to append the build information to the `getStudies()` table (default FALSE)
`studyId`	`character(1)` Indicates the "studyId" as taken from `getStudies`
`projection`	`character(1)` (default: "SUMMARY") Specify the projection type for data retrieval for details see API documentation
`molecularProfileIds`	`character()` A vector of molecular profile IDs

`entrezGeneIds`	`numeric()` A vector indicating entrez gene IDs
`sampleIds`	`character()` Sample identifiers
`sampleListId`	`character(1)` A sample list identifier as obtained from `sampleLists()`
`discreteCopyNumberEventType`	`character(1)` The copy number event type to filter on. Must be one of "HOMDEL_AND_AMP" (default), "HOMDEL", "AMP", "GAIN", "HETLOSS", "DIPLOID", or "ALL"
`keyword`	`character(1)` Keyword or pattern for searching through available operations
`sampleListIds`	`character()` A vector of 'sampleListId' as obtained from `sampleLists`
`genePanelId`	`character(1)` Identifies the gene panel, as obtained from the `genePanels` function
`molecularProfileId`	`character(1)` Indicates a molecular profile ID
`pageSize`	`numeric(1)` The number of rows in the table to return
`pageNumber`	`numeric(1)` The pagination page number
`...`	Additional arguments to lower level API functions
`by`	`character(1)` Either 'entrezGeneId' or 'hugoGeneSymbol' for row metadata (default: 'entrezGeneId')
`genes`	`character()` Either Entrez gene identifiers or Hugo gene symbols. When included, the 'by' argument indicates the type of identifier provided and 'genePanelId' is ignored. Preference is given to Entrez IDs due to faster query responses.

Value

- cBioPortal: An API object of class 'cBioPortal'
- cBioPortalData: A data object of class 'MultiAssayExperiment'

API Metadata

- getStudies: Obtain a table of studies and associated metadata and optionally include a `buildReport` status (default FALSE) for each study. When enabled, the 'api_build' and 'pack_build' columns will be added to the table and will show if `MultiAssayExperiment` objects can be generated for that particular study identifier (`studyId`). The 'api_build' column corresponds to datasets obtained with `cBioPortalData` and the 'pack_build' column corresponds to datsets loaded via `cBioDataPack`.
- searchOps - Search through API operations with a keyword
- sampleLists - obtain all `sampleListIds` for a particular `studyId`
- allSamples - obtain all samples within a particular `studyId`
- genePanels - Show all available gene panels
- geneTable - Get a table of all genes by 'entrezGeneId' and 'hugoGeneSymbol'
- queryGeneTable - Get a table for only the `genes` or `genePanelId` of interest. Gene inputs are identified with the `by` argument

Patient Data

- clinicalData - Obtain clinical data for a particular study identifier ('studyId')

Molecular Profiles

- molecularProfiles - Produce a molecular profiles dataset for a given study identifier ('studyId')

Molecular Data

- fetchData - A convenience function to download both mutation and molecular data with `molecularProfileId`, `entrezGeneIds`, and `sampleIds`

- mutationData - Produce a dataset of mutation data using `molecularProfileId`, `entrezGeneIds`, and `sampleIds`

- molecularData - Produce a dataset of molecular profile data based on `molecularProfileId`, `entrezGeneIds`, and `sampleIds`

Copy Number Data

- copyNumberData - Produce a dataset of copy number data based on `molecularProfileId`, `sampleListId`, `discreteCopyNumberEventType`, and `projection`

Sample Data

- samplesInSampleLists - get all samples associated with a 'sampleListId'

- getSampleInfo - Obtain sample metadata for a particular `studyId` or `sampleListId`

Gene Panels

- getGenePanels - Obtain the gene panel for a particular 'genePanelId'

- genePanelMolecular - get gene panel data for a particular `molecularProfileId` and either a vector of `sampleListId` or `sampleId`

- getGenePanelMolecular - get gene panel data for multiple `molecularProfileIds` and a vector of `sampleIds`

Genes

- getDataByGenes - Download data for a number of genes within `molecularProfileId` indicators, optionally a `sampleListId` can be provided.

Examples

```
cbio <- cBioPortal()

getStudies(api = cbio)

clinicalData(cbio, "acc_tcga")

molecularProfiles(cbio, "acc_tcga")

fetchData(
    api = cbio, studyId = "acc_tcga",
    molecularProfileIds = c(
        "acc_tcga_mutations", "acc_tcga_gistic", "acc_tcga_rppa"
    ),
    entrezGeneIds = 1:1000,
    sampleIds = c("TCGA-OR-A5J1-01", "TCGA-OR-A5J2-01")
)
mutationData(
    api = cbio,
    molecularProfileIds = "acc_tcga_mutations",
    entrezGeneIds = 1:1000,
    sampleIds = c("TCGA-OR-A5J1-01", "TCGA-OR-A5J2-01")
)

molecularData(
    api = cbio,
    molecularProfileIds = c("acc_tcga_rna_seq_v2_mrna", "acc_tcga_rppa"),
    entrezGeneIds = 1:100,
    sampleIds = c("TCGA-OR-A5J1-01", "TCGA-OR-A5J2-01")
)

## obtain molecularProfileId for discrete copy number alteration data
molecularProfiles(cbio, "acc_tcga") |>
    dplyr::filter(
        molecularAlterationType == "COPY_NUMBER_ALTERATION" &
        datatype == "DISCRETE"
    )

copyNumberData(
    api = cbio,
    molecularProfileIds = "acc_tcga_gistic",
    entrezGeneIds = 25,
    sampleListId = "acc_tcga_all"
)

searchOps(api = cbio, keyword = "molecular")

samplesInSampleLists(
    api = cbio,
    sampleListIds = c("acc_tcga_rppa", "acc_tcga_cnaseq")
)
```

```
sampleLists(api = cbio, studyId = "acc_tcga")

genePanels(cbio)

getGenePanel(cbio, "AmpliSeq")

queryGeneTable(api = cbio, by = "entrezGeneId", genes = 7157)

getDataByGenes(
    api = cbio,
    studyId = "acc_tcga",
    genes = 1,
    by = "entrezGeneId",
    molecularProfileIds = "acc_tcga_rna_seq_v2_mrna",
    sampleListId = "acc_tcga_rna_seq_v2_mrna"
)
```

`cBioPortal-class` *A class for representing the cBioPortal API protocol*

Description

The `cBioPortal` class is a representation of the `cBioPortal` API protocol that directly inherits from the `Service` class in the `AnVIL` package. For more information, see the AnVIL package.

Usage

```
## S4 method for signature 'cBioPortal'
operations(x, ..., .deprecated = FALSE)
```

Arguments

`x`	A AnVIL instance or API representation as given by the cBioPortal function.
`...`	additional arguments passed to methods or, for `operations,Service-method`, to the internal `get_operation()` function.
`.deprecated`	optional logical(1) include deprecated operations?

Details

This class takes the static API as provided at `https://www.cbioportal.org/api/v2/api-docs` and creates an R object with the help from underlying infrastructure (i.e., rapiclient and AnVIL) to give the user a unified representation of the API specification provided by the cBioPortal group. Users are not expected to interact with this class other than to use it as input to the functionality provided by the rest of the package.

Value

A `cBioPortal` class instance

Functions

- `operations(cBioPortal)`: List all the `operations` available with the cBioPortal API object, e.g., `api$operation`

See Also

cBioPortal, AnVIL

Examples

```
cBioPortal()
```

`cBioPortalData` *Download data from the cBioPortal API*

Description

Obtain a `MultiAssayExperiment` object for a particular gene panel, `studyId`, `molecularProfileIds`, and `sampleListIds` combination. Default `molecularProfileIds` and `sampleListIds` are set to NULL for including all data. This option is best for users who wish to obtain a section of the study data that pertains to a specific molecular profile and gene panel combination. For users looking to download the entire study data as provided by the `https://www.cbioportal.org/datasets`, refer to `cBioDataPack`.

Usage

```
cBioPortalData(
  api,
  studyId = NA_character_,
  genePanelId = NA_character_,
  genes = NA_character_,
  molecularProfileIds = NULL,
  sampleListId = NULL,
  sampleIds = NULL,
  by = c("entrezGeneId", "hugoGeneSymbol"),
  check_build = TRUE,
  ask = interactive()
)
```

Arguments

`api`	An API object of class `cBioPortal` from the `cBioPortal` function
`studyId`	`character(1)` Indicates the "studyId" as taken from `getStudies`
`genePanelId`	`character(1)` Identifies the gene panel, as obtained from the `genePanels` function

`genes`	`character()` Either Entrez gene identifiers or Hugo gene symbols. When included, the 'by' argument indicates the type of identifier provided and 'genePanelId' is ignored. Preference is given to Entrez IDs due to faster query responses.
`molecularProfileIds`	`character()` A vector of molecular profile IDs
`sampleListId`	`character(1)` A sample list identifier as obtained from `sampleLists()`
`sampleIds`	`character()` Sample identifiers
`by`	`character(1)` Either 'entrezGeneId' or 'hugoGeneSymbol' for row metadata (default: 'entrezGeneId')
`check_build`	logical(1L) Whether to check the build status of the `studyId` using an internal dataset. This argument should be set to `FALSE` if using alternative `hostnames`, e.g., 'pedcbioportal.kidsfirstdrc.org'
`ask`	`logical(1)` Whether to prompt the the user before downloading and loading study `MultiAssayExperiment` that is not currently building based on previous testing. Set to `interactive()` by default. In a non-interactive session, data download will be attempted; equivalent to `ask = FALSE`. The argument will also be used when a cache directory needs to be created when using `downloadStudy`.

Details

We are able to succesfully represent 98 percent of the study identifiers as `MultiAssayExperiment` objects as obtained via `cBioPortalData` with the `IMPACT341 genePanelId` as the example gene panel. Datasets that currently fail to import can be seen in the `getStudies(..., buildReport = TRUE)` dataset under the `"api_build"` column. Note that changes to the cBioPortal API may affect this rate at any time. If you encounter any issues, please open a GitHub issue at the `https://github.com/waldronlab/cBioPortalData/issues/` page with a fully reproducible example.

Value

A MultiAssayExperiment object

See Also

cBioDataPack, removeDataCache

Examples

```
cbio <- cBioPortal()

samps <- samplesInSampleLists(cbio, "acc_tcga_rppa")[[1]]

getGenePanelMolecular(
    cbio, molecularProfileIds = c("acc_tcga_rppa", "acc_tcga_linear_CNA"),
    samps
)

acc_tcga <- cBioPortalData(
    cbio, by = "hugoGeneSymbol",
    studyId = "acc_tcga",
```

```
    genePanelId = "AmpliSeq",
    molecularProfileIds =
        c("acc_tcga_rppa", "acc_tcga_linear_CNA", "acc_tcga_mutations")
)
```

`downloadStudy` *Manually download, untar, and load study tarballs*

Description

Note that these functions should be used when a particular study is *not* currently available as a `MultiAssayExperiment` representation. Otherwise, use `cBioDataPack`. Provide a `cancer_study_id` from `getStudies` and retrieve the study tarball from the cBio Genomics Portal. These functions are used by `cBioDataPack` under the hood to download,untar, and load the tarball datasets with caching. As stated in `?cBioDataPack`, not all studies are currently working as `MultiAssayExperiment` objects. As of July 2020, about ~80% of datasets can be successfully imported into the `MultiAssayExperiment` data class. Please open an issue if you would like the team to prioritize a study. You may also check `getStudies(buildReport = TRUE)$pack_build` for the current status.

Usage

```
downloadStudy(
  cancer_study_id,
  use_cache = TRUE,
  force = FALSE,
  url_location = getOption("cBio_URL", .url_location),
  ask = interactive()
)

untarStudy(cancer_study_file, exdir = tempdir())

loadStudy(
  filepath,
 names.field = c("Hugo_Symbol", "Entrez_Gene_Id", "Gene", "Composite.Element.REF"),
  cleanup = TRUE
)
```

Arguments

`cancer_study_id`	`character(1)` The study identifier from cBioPortal as seen in the dataset links at `https://www.cbioportal.org/datasets`.
`use_cache`	`logical(1)` (default TRUE) create the default cache location and use it to track downloaded data. If data found in the cache, data will not be re-downloaded. A path can also be provided to data cache location.

`force`	`logical(1)` (default FALSE) whether to force re-download data from remote location
`url_location`	`character(1)` (default "https://cbioportal-datahub.s3.amazonaws.com") the URL location for downloading packaged data. Can be set using the 'cBio_URL' option (see `?cBioDataPack` for more details)
`ask`	`logical(1)` Whether to prompt the the user before downloading and loading study `MultiAssayExperiment` that is not currently building based on previous testing. Set to `interactive()` by default. In a non-interactive session, data download will be attempted; equivalent to `ask = FALSE`. The argument will also be used when a cache directory needs to be created when using `downloadStudy`.
`cancer_study_file`	`character(1)` indicates the on-disk location of the downloaded tarball
`exdir`	`character(1)` indicates the folder location to *put* the contents of the tarball (default `tempdir()`; see also `?untar`)
`filepath`	`character(1)` indicates the folder location where the contents of the tarball are *located* (usually the same as `exdir`)
`names.field`	`character()` Possible column names for the column that will used to label ranges for data such as mutations or copy number (defaults: "Hugo_Symbol", "Entrez_Gene_Id", "Gene", and "Composite.Element.REF"). Values are cycled through and eliminated when no data present, or duplicates are found. Values in the corresponding column must be unique in each row.
`cleanup`	`logical(1)` whether to delete the `untar`-red contents from the `exdir` folder (default TRUE)

Details

When attempting to load a dataset using `loadStudy`, note that the `cleanup` argument is set to `TRUE` by default. Change the argument to `FALSE` if you would like to keep the untarred data in the `exdir` location. `downloadStudy` and `untarStudy` are not affected by this change. The tarball of the downloaded data is cached via `BiocFileCache` when `use_cache` is `TRUE`.

Value

- downloadStudy - The file location of the data tarball
- untarStudy - The directory location of the contents
- loadStudy - A MultiAssayExperiment-class object

See Also

cBioDataPack, MultiAssayExperiment

Examples

```
acc_file <- downloadStudy("acc_tcga")
acc_file

file_dir <- untarStudy(acc_file, tempdir())
```

```
file_dir

loadStudy(file_dir)
```

Index