Package 'cellxgenedp'

April 10, 2023

Title Discover and Access Single Cell Data Sets in the cellxgene Data Portal

Version 1.2.2

Description The cellxgene data portal (https://cellxgene.cziscience.com/) provides a graphical user interface to collections of single-cell sequence data processed in standard ways to 'count matrix' summaries. The cellxgenedp package provides an alternative, R-based inteface, allowind data discovery, viewing, and downloading.

License Artistic-2.0

Encoding UTF-8

Collate db.R collections.R datasets.R files.R facets.R keys.R cellxgene.R utilities.R cxg.R

Depends dplyr

Imports httr, curl, jsonlite, utils, tools, shiny, DT, rjsoncons

Suggests zellkonverter, SingleCellExperiment, HDF5Array, BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0), mockery

biocViews SingleCell, DataImport, ThirdPartyClient

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.1

VignetteBuilder knitr

Config/testthat/edition 3

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

git_branch RELEASE_3_16

git_last_commit 9687a46

git_last_commit_date 2023-01-31

Date/Publication 2023-04-10

Author Martin Morgan [aut, cre] (<https://orcid.org/0000-0002-5874-8148>), Kayla Interdonato [aut]

Maintainer Martin Morgan <mtmorgan.bioc@gmail.com>

R topics documented:

`collections` *Query cellxgene collections, datasets, and files*

Description

`files_download()` retrieves one or more cellxgene files to a cache on the local system.

Usage

```
collections(cellxgene_db = db())

datasets(cellxgene_db = db())

datasets_visualize(tbl)

files(cellxgene_db = db())

files_download(tbl, dry.run = TRUE, cache.path = .cellxgene_cache_path())
```

Arguments

`cellxgene_db`	an optional 'cellxgene_db' object, as returned by `db()`.
`tbl`	a `tibble()` typically derived from `datasets(db)` or `files(db)` and containing columns `dataset_id` (for `datasets_visualize()`), or columns `dataset_id`, `file_id`, and `filetype` (for `files_download()`).
`dry.run`	logical(1) indicating whether the (often large) file(s) in `tbl` should be downloaded to a local cache. Files are not downloaded when `dry.run = TRUE` (default).
`cache.path`	character(1) directory in which to cache downloaded files. The directory must already exist. The default is `tools::R_user_dir("cellxgenedp", "cache")`, a package-specific path in the user home directory.

Value

Each function returns a tibble describing the corresponding component of the database.

`files_download()` returns a character() vector of paths to the local files.

Examples

```
db <- db()

collections(db)

collections(db) |>
    dplyr::glimpse()

datasets(db) |>
    dplyr::glimpse()


## visualize the first dataset
datasets(db) |>
    dplyr::slice(1) |>
    datasets_visualize()

files(db) |>
    dplyr::glimpse()

## Not run:
files(db) |>
    dplyr::slice(1) |>
    files_download(dry.run = FALSE)

## End(Not run)
```

---

cxg *Shiny application for discovering, viewing, and downloading cellxgene data*

Description

Shiny application for discovering, viewing, and downloading cellxgene data

Usage

```
cxg(as = c("tibble", "sce"))
```

Arguments

as	character(1) Return value when quiting the shiny application. "tibble" returns a tibble describing selected datasets (including the location on disk of the downloaded file). "sce" returns a list of dataset files imported to R as SingleCellExperiment objects.

Value

cxg() returns either a tibble describing datasets selected in the shiny application, or a list of datasets imported into R as SingleCellExperiment objects.

Examples

```
cxg()
```

---

db *Retrieve updated cellxgene database metadata*

Description

Retrieve updated cellxgene database metadata

Usage

```
db(overwrite = .db_online() && .db_first())
```

Arguments

overwrite	logical(1) indicating whether the database of collections should be updated from the internet (the default, when internet is available and, in an interactive session, the user requests the update), or read from disk (assuming previous successful access to the internet). overwrite = FALSE might be useful for reproducibility, testing, or when working in an environment with restricted internet access.

Details

The database is retrieved from the cellxgene data portal web site. 'collections' metadata are retrieved on each call; metadata on each collection is cached locally for re-use.

Value

db() returns an object of class 'cellxgene_db', summarizing available collections, datasets, and files.

Examples

```
db()
```

FACETS *Facets available for querying cellxgene data*

Description

FACETS is a character vector of common fields used to subset cellxgene data.

facets() is used to query the cellxgene database for current values of one or all facets.

facets_filter() provides a convenient way to filter facets based on label or ontology term.

Usage

```
FACETS

facets(cellxgene_db = db(), facets = FACETS)

facets_filter(facet, key = c("label", "ontology_term_id"), value, exact = TRUE)
```

Arguments

cellxgene_db	an (optional) cellxgene_db object, as returned by db().
facets	a character() vector corersponding to one of the facets in FACETS.
facet	the column containing faceted information, e.g., sex in datasets(db).
key	character(1) identifying whether value is a label or ontology_term_id.
value	character() value of the label or ontology term to filter on. The value may be a vector with length(value) > 0 for exact matchs (exact = TRUE, default), or a character(1) regular expression.
exact	logical(1) whether values match exactly (default, TRUE) or as a regular expression (FALSE).

Format

FACETS is an object of class character of length 8.

Value

facets() returns a tibble with columns facet, label, ontology_term_id, and n, the number of times the facet label is used in the database.

facets_filter() returns a logical vector with length equal to the length (number of rows) of facet, with TRUE indicating that the value of key is present in the dataset.

Examples

```
f <- facets()

## levels of each facet
f |>
    dplyr::count(facet)

## same as facets(, facets = "organism")
f |>
    dplyr::filter(facet == "organism")

db <- db()
ds <- datasets(db)

## datasets with African American females
ds |>
    dplyr::filter(
        facets_filter(self_reported_ethnicity, "label", "African American"),
        facets_filter(sex, "label", "female")
    )

## datasets with non-European, known ethnicity
facets(db, "self_reported_ethnicity")
ds |>
    dplyr::filter(
        !facets_filter(
            self_reported_ethnicity, "label", c("European", "na", "unknown")
        )
    )
```

Index