

Package ‘cellxgenedp’

April 5, 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 interface, allowing 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 2023-04-05

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	2
cxg	3
db	4
FACETS	5
Index	7

collections	<i>Query cellxgene collections, datasets, and files</i>
-------------	---

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

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

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). <code>overwrite = FALSE</code> 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

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

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

* **datasets**

FACETS, [5](#)

`collections`, [2](#)

`cxg`, [3](#)

`datasets(collections)`, [2](#)

`datasets_visualize(collections)`, [2](#)

`db`, [4](#)

FACETS, [5](#)

`facets(FACETS)`, [5](#)

`facets_filter(FACETS)`, [5](#)

`files(collections)`, [2](#)

`files_download(collections)`, [2](#)