Package ‘UCSC.utils’

June 26, 2025

Title Low-level utilities to retrieve data from the UCSC Genome Browser

Description A set of low-level utilities to retrieve data from the UCSC Genome Browser. Most functions in the package access the data via the UCSC REST API but some of them query the UCSC MySQL server directly. Note that the primary purpose of the package is to support higher-level functionalities implemented in downstream packages like GenomeInfoDb or txdbmaker.

biocViews Infrastructure, GenomeAssembly, Annotation, GenomeAnnotation, DataImport

URL https://bioconductor.org/packages/UCSC.utils

BugReports https://github.com/Bioconductor/UCSC.utils/issues

Version 1.4.0

License Artistic-2.0

Encoding UTF-8

Imports methods, stats, httr, jsonlite, S4Vectors

Suggests DBI, RMariaDB, GenomeInfoDb, testthat, knitr, rmarkdown, BiocStyle

VignetteBuilder knitr

Collate 00utils.R UCSC.api.url.R REST_API.R list_UCSC_genomes.R get_UCSC_chrom_sizes.R list_UCSC_tracks.R fetch_UCSC_track_data.R UCSC_dbselect.R zzz.R

git_url https://git.bioconductor.org/packages/UCSC.utils

git_branch RELEASE_3_21

git_last_commit 8ff2504

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-06-25

Author Hervé Pagès [aut, cre]

Maintainer Hervé Pagès <hpages.on.github@gmail.com>

Contents

fetch_UCSC_track_data 2
get_UCSC_chrom_sizes 3
list_UCSC_genomes 4
list_UCSC_tracks 6
UCSC.api.url 7
UCSC_dbselect 8

Index **11**

`fetch_UCSC_track_data` *Fetch UCSC track data*

Description

Fetch the track data for a given UCSC genome/track.

Usage

```
fetch_UCSC_track_data(genome, primary_table, api.url=UCSC.api.url())
```

Arguments

`genome`	A single string specifying the name of a UCSC genome e.g. `"hs1"`, `"danRer11"`, or `"wuhCor1"`. See ?`list_UCSC_genomes` for how to get the list of valid UCSC genome names.
`primary_table`	A single string specifying the name of the primary table associated with the track from which to fetch the data. See ?`list_UCSC_tracks` for how to get the list of tracks and associated primary tables for a given UCSC genome.
`api.url`	The URL of the UCSC API. By default, the URL returned by `UCSC.api.url()` is used. Note that what `UCSC.api.url()` returns is controlled by a global option. See ?`UCSC.api.url` for more information.

Value

A data frame.

See Also

- `list_UCSC_genomes` to get the list of UCSC genomes.
- `list_UCSC_tracks` to get the list of tracks and associated primary tables for a given UCSC genome.
- `UCSC_dbselect` for a more efficient and more flexible way to retrieve data directly from the UCSC MariaDB server.

Examples

```
gorGor6_gap_data <- fetch_UCSC_track_data("gorGor6", "gap")
head(gorGor6_gap_data)

## --- Comparison with UCSC_dbselect() ---

gorGor6_gap_data2 <- UCSC_dbselect("gorGor6", "gap")

## Easy sanity checks.
stopifnot(
  identical(dim(gorGor6_gap_data), dim(gorGor6_gap_data2)),
  identical(colnames(gorGor6_gap_data), colnames(gorGor6_gap_data2))
)

## But the two data frames are not identical:
identical(gorGor6_gap_data, gorGor6_gap_data2)  # FALSE!

## However, they have the same content. Ony reason they're not
## identical is because their rows are not in the same order.
## Let's reorder the rows in the two data frames by genomic location,
## that is, first by chromosome, then by chromStart, and finally
## by chromEnd:
sort_rows <- function(gap_data) {
  oo <- order(gap_data$chrom, gap_data$chromStart, gap_data$chromEnd)
  gap_data <- gap_data[oo, ]
  rownames(gap_data) <- NULL
  gap_data
}

gorGor6_gap_data <- sort_rows(gorGor6_gap_data)
gorGor6_gap_data2 <- sort_rows(gorGor6_gap_data2)

## Now the two data frames are identical:
stopifnot(identical(gorGor6_gap_data, gorGor6_gap_data2))
```

---

get_UCSC_chrom_sizes *List UCSC chromosome sizes*

Description

Get the chromosome sizes of a given UCSC genome.

Usage

```
get_UCSC_chrom_sizes(genome, api.url=UCSC.api.url(), recache=FALSE)
```

Arguments

genome — A single string specifying the name of a UCSC genome e.g. "hs1", "mm39", or "sacCer3". See ?list_UCSC_genomes for how to get the list of valid UCSC genome names.

api.url — The URL of the UCSC API. By default, the URL returned by UCSC.api.url() is used. Note that what UCSC.api.url() returns is controlled by a global option. See ?UCSC.api.url for more information.

recache — get_UCSC_chrom_sizes() uses a cache mechanism so the information retrieved for a given genome only gets downloaded once during the current R session (note that the caching is done in memory so cached information does NOT persist across sessions). Setting recache to TRUE forces a new download (and re-caching) of the chromosome sizes for the specified genome.

Value

A named numeric vector. The names on the vector are the UCSC chromosomes/sequences. The vector values are the corresponding lengths.

Note that the vector is not sorted in any particular order. In particular there's not guarantee that the chromosomes will precede the scaffolds.

See Also

- list_UCSC_genomes to get the list of UCSC genomes.
- The Seqinfo constructor function in the **GenomeInfoDb** package for an alternate (higher level) way of retrieving the chromosome information of a given NCBI assembly or UCSC genome.
- list_UCSC_tracks to get the list of tracks and associated primary tables for a given UCSC genome.
- UCSC.api.url for how to use an alternative UCSC API URL by default.

Examples

```
get_UCSC_chrom_sizes("ce2")

get_UCSC_chrom_sizes("hg38")
```

---

list_UCSC_genomes *List UCSC genomes*

Description

Get the list of UCSC genomes.

Usage

```
list_UCSC_genomes(organism=NA, api.url=UCSC.api.url(), recache=FALSE)

get_organism_for_UCSC_genome(genome, api.url=UCSC.api.url(), recache=FALSE)
```

Arguments

organism	By default all UCSC genomes are returned (in a data frame). When organism is specified, list_UCSC_genomes() will only return the rows associated with the specified organism. organism must be supplied as a single string that will be used to perform a search (with grep()) on the organism and common_name columns of the data frame to return. The search is case-insensitive.
api.url	The URL of the UCSC API. By default, the URL returned by UCSC.api.url() is used. Note that what UCSC.api.url() returns is controlled by a global option. See ?UCSC.api.url for more information.
recache	list_UCSC_genomes() uses a cache mechanism so the list of genomes only gets downloaded once during the current R session (note that the caching is done in memory so cached information does NOT persist across sessions). Setting recache to TRUE forces a new download (and recaching) of the list of genomes. In the case of get_organism_for_UCSC_genome(), the supplied recache value is just passed down to the internal call to list_UCSC_genomes().
genome	A character vector of valid UCSC genomes e.g. "hg38", "mm39", or "sacCer3".

Value

For list_UCSC_genomes: A data frame with 1 row per genome and 5 columns: organism, genome, common_name, tax_id, description.

For get_organism_for_UCSC_genome: A named character of the same length as the input containing the scientific names of the organisms associated with the supplied UCSC genomes.

See Also

- https://genome.ucsc.edu/FAQ/FAQreleases.html for the online HTML page that lists all UCSC genome releases, including archived ones.
- registered_UCSC_genomes in the **GenomeInfoDb** package for a similar function that returns only UCSC genomes registered in the **GenomeInfoDb** package.
- get_UCSC_chrom_sizes to get the chromosome sizes of a given UCSC genome.
- list_UCSC_tracks to get the list of tracks and associated primary tables for a given UCSC genome.
- UCSC.api.url for how to use an alternative UCSC API URL by default.

Examples

```
list_UCSC_genomes("human")

list_UCSC_genomes("pacos")

get_organism_for_UCSC_genome(c("ce11", "xenTro10", "mpxvRivers"))
```

list_UCSC_tracks *List UCSC tracks and associated primary tables*

Description

Get the list of tracks and associated primary tables for a given UCSC genome.

Usage

```
list_UCSC_tracks(genome, group=NULL,
                 api.url=UCSC.api.url(), recache=FALSE)
```

Arguments

genome	A single string specifying the name of a UCSC genome e.g. "hg38", "mm39", or "sacCer3". See ?list_UCSC_genomes for how to get the list of valid UCSC genome names.
group	NULL or a single string specifying the group of tracks to return. By default, all tracks are returned. Passing group=NA is accepted and will return only rows associated with tracks that don't belong to any group.
api.url	The URL of the UCSC API. By default, the URL returned by UCSC.api.url() is used. Note that what UCSC.api.url() returns is controlled by a global option. See ?UCSC.api.url for more information.
recache	list_UCSC_tracks() uses a cache mechanism so the information retrieved for a given genome only gets downloaded once during the current R session (note that the caching is done in memory so cached information does NOT persist across sessions). Setting recache to TRUE forces a new download (and re-caching) of the list of tracks for the specified genome.

Value

A data frame with 1 row per track and 5 columns: track, primary_table, type, group, composite_track.

Note that columns group and composite_track can contain NAs.

See Also

- list_UCSC_genomes to get the list of UCSC genomes.
- fetch_UCSC_track_data to fetch the track data for a given UCSC genome/track.
- get_UCSC_chrom_sizes to get the chromosome sizes of a given UCSC genome.
- UCSC.api.url for how to use an alternative UCSC API URL by default.

Examples

```
## List all tracks for ce2 genome:
list_UCSC_tracks("ce2")

## List tracks in the "rna" group only:
list_UCSC_tracks("ce2", group="rna")

## Note that some tracks don't belong to any group:
list_UCSC_tracks("hg38", group=NA)
```

`UCSC.api.url` *Get or set the default UCSC API URL*

Description

Convenience helper for getting or setting global option `UCSC.api.url`.

Usage

```
UCSC.api.url(api.url=NULL)
```

Arguments

`api.url` A single string containing the URL to the alternative UCSC API to use by default.

For convenience, `api.url` can also be one of the following aliases:

- `"primary"`: alias for `"https://api.genome.ucsc.edu"` (primary URL, US West Coast);
- `""`: same as `"primary"`;
- `"europe"`: alias for `"https://genome-euro.ucsc.edu/cgi-bin/hubApi"` (Europe mirror);
- `"asia"`: alias for `"https://genome-asia.ucsc.edu/cgi-bin/hubApi"` (Asia mirror).

Details

Various functions in the **UCSC.utils** package query the UCSC REST API. This is the case for example for `list_UCSC_genomes`, `get_UCSC_chrom_sizes`, `list_UCSC_tracks` and more.

Global option `UCSC.api.url` controls the UCSC API URL that these functions use y default. The option is set to `"https://api.genome.ucsc.edu"` (primary URL, US West Coast) at package startup.

`UCSC.api.url()` and `UCSC.api.url(some_url)` are provided as convenient ways of doing `getOption("UCSC.api.url")` and `options(UCSC.api.url=some_url)`, respectively.

Value

When called with no argument, `UCSC.api.url()` returns `getOption("UCSC.api.url")`.

When passed an URL, `UCSC.api.url(some_url)` returns the *previous* URL, that is, the UCSC API URL that was previously used by default. Note that the previous URL is returned invisibly.

See Also

- https://genome.ucsc.edu/goldenPath/help/api.html#Mirrors a list of alternative UCSC API URLs.
- `list_UCSC_genomes` to get the list of UCSC genomes.

Examples

```
UCSC.api.url()  # current default value of the UCSC API URL
get_UCSC_chrom_sizes("ce11", recache=TRUE)

## Temporarily use the mirror in Europe:
previous_url <- UCSC.api.url("europe")
UCSC.api.url()  # new default value of the UCSC API URL

get_UCSC_chrom_sizes("ce11", recache=TRUE)

## Restore previous default value:
UCSC.api.url(previous_url)
UCSC.api.url()
```

UCSC_dbselect *Conveniently retrieve data from the UCSC MariaDB server*

Description

A convenience utility to retrieve data from the UCSC MariaDB server via simple SQL `SELECT` statements.

Requires the **RMariaDB** package!

Usage

```
UCSC_dbselect(dbname, from, columns=NULL, where=NULL, MoreSQL=NULL,
              host="genome-mysql.soe.ucsc.edu", port=3306)
```

Arguments

`dbname`	A single string specifying the name of the database to connect to. This is usually the name of a valid UCSC genome e.g. `"hg38"`, `"mm39"`, or `"sacCer3"`. See `?list_UCSC_genomes` for how to get the list of valid UCSC genome names.

from	A single string specifying the *input data*, that is, the FROM clause of the SELECT statement. This is typically the name of an SQL table, or a join-clause, or a sub-query.
columns	NULL (the default), or a character vector specifying the columns to return. By default all columns are returned.
where	NULL (the default), or a single string specifying the WHERE clause of the SELECT statement.
MoreSQL	NULL (the default), or a character vector containing additional SQL clauses e.g. GROUP BY, ORDER BY, and/or LIMIT clauses.
host	A single string specifying the name of the server's host
port	The TCP/IP port to use to connect to the server.

Details

UCSC_dbselect is an alternative to fetch_UCSC_track_data that is more efficient and gives the user more control on what data to retrieve exactly from the server.

However, the downside is that UCSC_dbselect does not work with all tracks! This is because, not all track data are stored in a database: some tracks are actually file-based (e.g. bigBed tracks). See below for an example.

Value

A data frame.

See Also

- list_UCSC_genomes to get the list of UCSC genomes.
- list_UCSC_tracks to get the list of tracks and associated primary tables for a given UCSC genome.
- fetch_UCSC_track_data to fetch the track data for a given UCSC genome/track.
- https://genome.ucsc.edu/goldenpath/help/mysql.html for more information about downloading data from UCSC MariaDB databases.

Examples

```
### Retrieve full "ncbiGene" table ("NCBI Genes" track) for eboVir3:
UCSC_dbselect("eboVir3", "ncbiGene")

### Retrieve subset of "sgpGene" table ("SGP Genes" track) for hg38:
UCSC_dbselect("hg38", "sgpGene", where="chrom='chrM'")

### Retrieve subset of "gap" table for gorGor6:
columns <- c("chrom", "chromStart", "chromEnd", "type")
where <- "chrom='chrX' AND type='contig'"
UCSC_dbselect("gorGor6", "gap", columns=columns, where=where)

### With a LEFT JOIN and a LIMIT clause:
from <- paste("ncbiRefSeq LEFT JOIN ncbiRefSeqLink",
```

```
              "ON ncbiRefSeq.name=ncbiRefSeqLink.id")
res <- UCSC_dbselect("sacCer3", from, MoreSQL="LIMIT 5")
dim(res)

## WARNING: UCSC_dbselect() does not work with all tracks!
## For example it does not work with bigBed tracks:
subset(list_UCSC_tracks("hg38", group="genes"), grepl("bigBed 9", type))
## Not run:
UCSC_dbselect("hg38", "crisprAllTargets")  # table doesn't exist!

## End(Not run)
```

Index

∗ **manip**
    fetch_UCSC_track_data, 2
    get_UCSC_chrom_sizes, 3
    list_UCSC_genomes, 4
    list_UCSC_tracks, 6
    UCSC_dbselect, 8
∗ **utilities**
    UCSC.api.url, 7

fetch_UCSC_track_data, 2, *6*, *9*

get_organism_for_UCSC_genome
    *(*list_UCSC_genomes*)*, 4
get_UCSC_chrom_sizes, 3, *5–7*

list_UCSC_genomes, *2*, *4*, 4, *6–9*
list_UCSC_tracks, *2*, *4*, *5*, 6, *7*, *9*

registered_UCSC_genomes, *5*

Seqinfo, *4*

UCSC.api.url, *2*, *4–6*, 7
UCSC_dbselect, *2*, 8