Package 'biomartr'

February 23, 2022

Title Genomic Data Retrieval

Version 1.0.2

Description Perform large scale genomic data retrieval and functional annotation retrieval. This package aims to provide users with a standardized
way to automate genome, proteome, 'RNA', coding sequence ('CDS'), 'GFF', and metagenome retrieval from 'NCBI RefSeq', 'NCBI Genbank', 'ENSEMBL',
and 'UniProt' databases. Furthermore, an interface to the 'BioMart' database (Smedley et al. (2009) <doi:10.1186/1471-2164-10-22>) allows users to retrieve
functional annotation for genomic loci. In addition, users can download entire databases such as 'NCBI RefSeq' (Pruitt et al. (2007) <doi:10.1093/nar/gkl842>), 'NCBI nr',
'NCBI nt', 'NCBI Genbank' (Benson et al. (2013) <doi:10.1093/nar/gks1195>), etc. with only one command.

VignetteBuilder knitr

NeedsCompilation yes

Depends R (>= 3.1.1)

Imports biomaRt, Biostrings, curl, tibble, jsonlite, data.table (>= 1.9.4), dplyr (>= 0.3.0), readr (>= 1.4.0), downloader (>= 0.3), RCurl (>= 1.95-4.5), XML (>= 3.98-1.1), httr (>= 0.6.1), stringr (>= 0.6.2), purrr, R.utils, philentropy, withr, fs

Suggests knitr (>= 1.6), rmarkdown (>= 0.3.3), devtools (>= 1.6.1), testthat, seqinr, magrittr

License GPL-2

URL https://docs.ropensci.org/biomartr/, https://github.com/ropensci/biomartr

BugReports https://github.com/ropensci/biomartr/issues

RoxygenNote 7.1.2

Encoding UTF-8

X-schema.org-keywords BioMart, genomic-data-retrieval, annotation-retrieval, database-retrieval, NCBI, ENSEMBL, biological-data-retrieval

X-schema.org-applicationCategory Data Access

X-schema.org-isPartof ``ropensci.org''

Author Hajk-Georg Drost [aut, cre] (<https://orcid.org/0000-0002-1567-306X>)

Maintainer Hajk-Georg Drost <hajk-georg.drost@tuebingen.mpg.de>

Repository CRAN

Date/Publication 2022-02-23 13:50:09 UTC

R topics documented:

biomartr-package . . . 3
biomart . . . 4
check_annotation_biomartr . . . 5
clean.retrieval . . . 6
download.database . . . 7
download.database.all . . . 8
getAssemblyStats . . . 9
getAttributes . . . 11
getCDS . . . 12
getCDSSet . . . 13
getCollection . . . 15
getCollectionSet . . . 17
getDatasets . . . 19
getENSEMBL.gtf . . . 20
getENSEMBL.Seq . . . 21
getENSEMBLGENOMESInfo . . . 22
getENSEMBLInfo . . . 22
getFilters . . . 23
getGenome . . . 24
getGENOMEREPORT . . . 26
getGenomeSet . . . 26
getGFF . . . 28
getGFFSet . . . 30
getGO . . . 32
getGroups . . . 33
getGTF . . . 34
getKingdomAssemblySummary . . . 36
getKingdoms . . . 36
getMarts . . . 37
getMetaGenomeAnnotations . . . 38
getMetaGenomes . . . 39
getMetaGenomeSummary . . . 40
getProteome . . . 40
getProteomeSet . . . 42
getReleases . . . 44
getRepeatMasker . . . 45
getRNA . . . 46
getRNASet . . . 48

getSummaryFile . . . 49
is.genome.available . . . 50
listDatabases . . . 51
listGenomes . . . 52
listGroups . . . 54
listKingdoms . . . 55
listMetaGenomes . . . 56
meta.retrieval . . . 57
meta.retrieval.all . . . 60
organismAttributes . . . 61
organismBM . . . 63
organismFilters . . . 64
read_assemblystats . . . 65
read_cds . . . 66
read_genome . . . 67
read_gff . . . 68
read_proteome . . . 69
read_rm . . . 70
read_rna . . . 70
refseqOrganisms . . . 71
summary_cds . . . 72
summary_genome . . . 72

Index **74**

`biomartr-package` *Genomic Data Retrieval*

Description

This package interacts with a suite of web Application Programming Interfaces and FTP sites to perform automated genomic data retieval and annotation information retrieval.

About

To automate the retrieval process on a meta-genomic scale, this package provides useful interface functions for genomic sequence retrieval and functional annotation retrieval. The major aim of `biomartr` is to facilitate computational reproducibility and large-scale handling of genomic data for (meta-)genomic analyses.

In detail, `biomartr` aims to provide users with an easy to use framework to obtain genome, proteome, CDS, GFF (annotation), genome assembly quality, and metagenome project data. Furthermore, an interface to the Ensembl Biomart database allows users to retrieve functional annotation for genomic loci. Users can download entire databases such as

- `NCBI RefSeq`
- `NCBI nr`
- `NCBI nt`

- NCBI Genbank
- NCBI nt
- Ensembl
- Ensembl Genomes
- UniProt

Author(s)

Hajk-Georg Drost <hajk-georg.drost@tuebingen.mpg.de>

`biomart` *Main BioMart Query Function*

Description

This function takes a set of gene ids and the biomart specifications and performs a biomart query for the given set of gene ids.

Usage

```
biomart(genes, mart, dataset, attributes, filters, ...)
```

Arguments

`genes`	a character vector storing the gene ids of a organisms of interest to be queried against BioMart.
`mart`	a character string specifying the mart to be used. Users can obtain available marts using `getMarts`.
`dataset`	a character string specifying the dataset within the mart to be used, e.g. `dataset = "hsapiens_gene_ensembl"`.
`attributes`	a character vector specifying the attributes that shall be used, e.g. `attributes = c("start_position","end_position","description")`.
`filters`	a character vector specifying the filter (query key) for the BioMart query, e.g. `filter = "ensembl_gene_id"`.
`...`	additional parameters for the `getBM` function.

Details

This function is the main query function of the biomartr package.

It enables to fastly access annotations of a given gene set based on the **biomaRt** package implemented by Steffen Durinck et al.

Value

A data.table storing the initial query gene vector in the first column, the output gene vector in the second column, and all attributes in the following columns.

Author(s)

Hajk-Georg Drost

See Also

organismFilters, organismBM, listAttributes, getBM

Examples

```
## Not run:
# 1) select a mart
getMarts()

# we will select mart 'plants_mart' and search for available datasets
getDatasets(mart = "plants_mart")

# we choose dataset 'athaliana_eg_gene' and run biomart()
# using mart: 'plants_mart', dataset: "athaliana_eg_gene"
# attributes: c("start_position","end_position","description")
# for an example gene set of Arabidopsis thaliana:
# c("AT1G06090", "AT1G06100", "AT1G06110", "AT1G06120",
#    "AT1G06130", "AT1G06200")

biomart(genes      = c("AT1G06090", "AT1G06100",
                       "AT1G06110", "AT1G06120",
                       "AT1G06130", "AT1G06200"),
        mart       = "plants_mart",
        dataset    = "athaliana_eg_gene",
        attributes = c("start_position","end_position","description"),
        filters    = "ensembl_gene_id")

## End(Not run)
```

---

check_annotation_biomartr

Check whether an annotation file contains outlier lines

Description

Some annotation files include lines with character lengths greater than 65000. This causes problems when trying to import such annotation files into R using `import`. To overcome this issue, this function screens for such lines in a given annotation file and removes these lines so that `import` can handle the file.

Usage

```
check_annotation_biomartr(annotation_file, remove_annotation_outliers = FALSE)
```

Arguments

annotation_file
: a file path to the annotation file.

remove_annotation_outliers
: shall outlier lines be removed from the input annotation_file? If yes, then the initial annotation_file will be overwritten and the removed outlier lines will be stored at tempdir for further exploration.

Author(s)

Hajk-Georg Drost

Examples

```
## Not run:
# download an example annotation file from NCBI RefSeq
Ath_path <- biomartr::getGFF(organism = "Arabidopsis thaliana")
# run annotation file check on the downloaded file
biomartr::check_annotation_biomartr(Ath_path)
# several outlier lines were detected, thus we re-run the
# function using 'remove_annotation_outliers = TRUE'
# to remove the outliers and overwrite the file
biomartr::check_annotation_biomartr(Ath_path, remove_annotation_outliers = TRUE)

## End(Not run)
```

---

clean.retrieval *Format* meta.retrieval *output*

Description

Process the output of meta.retrieval by first un-zipping downloaded files and renaming them for more convenient downstream data analysis.

Usage

```
clean.retrieval(x, gunzip = TRUE)
```

Arguments

x
: a vector containing file paths to the output files generated by meta.retrieval.

gunzip
: a logical value indicating whether or not files should only be renamed (gunzip = FALSE) or renamed AND unzipped (gunzip).

Details

The output of meta.retrieval usually contains compressed sequence files and a naming convention based on the database the respective file was retrieved from (e.g. Saccharomyces_cerevisiae_cds_from_genomic_ref
This function helps to format the meta.retrieval output files by

- 1) Automatically uncompress all sequence files in the meta.retrieval output folder
- 2) Automatically rename files from e.g. Saccharomyces_cerevisiae_cds_from_genomic_refseq.fna.gz to Scerevisiae.fa. This allows more convenient downstream analyses and visualizations.

Author(s)

Hajk-Georg Drost

See Also

meta.retrieval

Examples

```
## Not run:
# The easiest way to use 'clean.retrieval()' in combination with
# 'meta.retrieval()' is to use the pipe operator from the 'magrittr' package
library(magrittr)
meta.retrieval(kingdom = "vertebrate_mammalian",
               db = "refseq",
               type = "genome") %>% clean.retrieval()

## End(Not run)
```

---

download.database *Download a NCBI Database to Your Local Hard Drive*

Description

This function allows users to download a database selected by listDatabases to their local hard drive.

Usage

```
download.database(db, path = "database")
```

Arguments

db	a character string specifying the database that shall be downloaded (selected from listDatabases).
path	a character string specifying the location (a folder) in which the corresponding database shall be stored. Default is path = "database". In case this folder does not exist yet, it will be created.

Details

This function downloads large databases to your hard drive. For this purpose a folder named `database` (default) is created and the correspondning database then stored in this folder.

Value

File path to the downloaded database file.

Author(s)

Hajk-Georg Drost

See Also

`download.database.all`, `listDatabases`

Examples

```
## Not run:
  # search for available NCBI nr databases
  listNCBIDatabases(db = "nr")
  # select NCBI nr version 27 =  "nr.27.tar.gz"
  # and download it to your hard drive
  # -> please note that large databases take some time for download!
  download.database(db = "nr.27.tar.gz")

## End(Not run)
```

`download.database.all` *Download all elements of an NCBI databse*

Description

The `download.database` functions allows users to retrieve individual packages of a NCBI database. This function is designed to retrieve the entire database selected by the users (hence all packages corresponding to this database).

Usage

```
download.database.all(db, path = NULL)
```

Arguments

`db`	a character string specifying the database that shall be downloaded (selected from `listDatabases`).
`path`	a character string specifying the location (a folder) in which the corresponding database shall be stored. In case this folder does not exist yet, it will be created.

Value

A character vector storing the file paths of the downloaded databases.

Author(s)

Hajk-Georg Drost

See Also

download.database, listNCBIDatabases

Examples

```
## Not run:
# search for available NCBI databases
  listNCBIDatabases(db = "all")
# choose database NCBI nr and download compelete database
  download.database.all(db = "nr", path = "nr")

## End(Not run)
```

---

getAssemblyStats *Genome Assembly Stats Retrieval*

Description

Main genome assembly stats retrieval function for an organism of interest. By specifying the scientific name of an organism of interest the corresponding genome assembly stats file storing the assembly statistics of the organism of interest can be downloaded and stored locally. Genome assembly stats files can be retrieved from several databases.

Usage

```
getAssemblyStats(
  db = "refseq",
  organism,
  reference = FALSE,
  type = "download",
  path = file.path("_ncbi_downloads", "genomeassembly_stats")
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank"

- db = "ensembl"

organism	a character string specifying the scientific name of the organism of interest, e.g. organism = "Homo sapiens".
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
type	shall only the file be retrieved (default) type = "download" or should the corresponding file be downloaded and subsequently be imported type = "import".
path	a character string specifying the location (a folder) in which the corresponding file shall be stored. Default is path = file.path("_ncbi_downloads","genomeassembly_stats").

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

to retrieve available scientific names of organisms and creates a directory '_ncbi_downloads/genomeassembly_stats' to store the Genome Assembly Stats of interest as text file for future processing. In case the corresponding fasta file already exists within the '_ncbi_downloads/genomeassembly_stats' folder and is accessible within the workspace, no download process will be performed.

An example genome assembly stats file can be found here: ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/405/GCF_000001405.36_GRCh38.p10/GCF_000001405.36_GRCh38.p10_assembly_stats.txt.

Value

File path to downloaded genome assembly stats file.

Author(s)

Hajk-Georg Drost

See Also

getGenome, getProteome, getCDS, getGFF, getRNA, meta.retrieval, read_assemblystats

Examples

```
## Not run:
# download the genome assembly stats file of Saccharomyces cerevisiae
# from NCBI RefSeq
# and store the corresponding genome file in
# '_ncbi_downloads/genomeassembly_stats'
file_path <- getAssemblyStats( db = "refseq",
                organism = "Saccharomyces cerevisiae",
                path = file.path("_ncbi_downloads","genomeassembly_stats"))
# import the raw file as it is downloaded
Scerevisiae.stats <- read_assemblystats(file_path, type = "raw")

# download the genome assembly stats file of Saccharomyces cerevisiae
```

```
# from NCBI RefSeq
# and import overall statistics of the genome assembly
Scerevisiae.stats.import <- getAssemblyStats( db = "refseq",
                organism = "Saccharomyces cerevisiae",
                type = "import",
                path = file.path("_ncbi_downloads","genomeassembly_stats"))

## End(Not run)
```

`getAttributes` *Retrieve All Available Attributes for a Specific Dataset*

Description

This function queries the BioMart Interface and returns a table storing all available attributes for a specific dataset.

Usage

```
getAttributes(mart, dataset)
```

Arguments

`mart`	a character string specifying the database (mart) for which datasets shall be listed.
`dataset`	a character string specifying the dataset for which attributes shall be listed.

Author(s)

Hajk-Georg Drost

See Also

`getMarts`, `getDatasets`, `getFilters`, `organismBM`, `organismFilters`, `organismAttributes`

Examples

```
## Not run:
# search for available datasets
getMarts()

# choose database (mart): ENSEMBL_MART_ENSEMBL
# and get a table of all available datasets from this BioMart database
head(getDatasets(mart = "ENSEMBL_MART_ENSEMBL"), 10)

# choose dataset: "hsapiens_gene_ensembl"
head(getAttributes(mart = "ENSEMBL_MART_ENSEMBL",
                   dataset = "hsapiens_gene_ensembl") , 5)

## End(Not run)
```

getCDS *Coding Sequence Retrieval*

Description

Main retrieval function for coding sequences (CDS) of an organism of interest. By specifying the scientific name of an organism of interest the corresponding fasta-file storing the CDS information for the organism of interest can be downloaded and stored locally. CDS files can be retrieved from several databases.

Usage

```
getCDS(
  db = "refseq",
  organism,
  reference = FALSE,
  release = NULL,
  gunzip = FALSE,
  path = file.path("_ncbi_downloads", "CDS")
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organism	there are three options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
gunzip	a logical value indicating whether or not files should be unzipped.
path	a character string specifying the location (a folder) in which the corresponding CDS file shall be stored. Default is path = file.path("_ncbi_downloads","CDS").

Value

File path to downloaded CDS file.

Author(s)

Hajk-Georg Drost

See Also

getGenome, getProteome, getGFF, getRNA, getRepeatMasker, getAssemblyStats, meta.retrieval, read_cds

Examples

```
## Not run:
# download the genome of Arabidopsis thaliana from refseq
# and store the corresponding genome CDS file in '_ncbi_downloads/CDS'
file_path <- getCDS( db       = "refseq",
             organism = "Arabidopsis thaliana",
             path     = file.path("_ncbi_downloads","CDS"))

Ath_CDS <- read_cds(file_path, format = "fasta")


## End(Not run)
```

---

getCDSSet *CDS retrieval of multiple species*

Description

Main CDS retrieval function for a set of organism of interest. By specifying the scientific names of the organisms of interest the corresponding fasta-files storing the CDS of the organisms of interest will be downloaded and stored locally. CDS files can be retrieved from several databases.

Usage

```
getCDSSet(
  db = "refseq",
  organisms,
  reference = FALSE,
  release = NULL,
  clean_retrieval = TRUE,
  gunzip = TRUE,
  update = FALSE,
  path = "set_CDS"
)
```

Arguments

db	a character string specifying the database from which the CDS shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organisms	a character vector storing the names of the organisms than shall be retrieved. There are three available options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a CDS shall be downloaded if it isn't marked in the database as either a reference CDS or a representative CDS.
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
clean_retrieval	logical value indicating whether or not downloaded files shall be renamed for more convenient downstream data analysis.
gunzip	a logical value indicating whether or not files should be unzipped.
update	a logical value indicating whether or not files that were already downloaded and are still present in the output folder shall be updated and re-loaded (update = TRUE or whether the existing file shall be retained update = FALSE (Default)).
path	a character string specifying the location (a folder) in which the corresponding CDSs shall be stored. Default is path = "set_CDS".

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory 'set_CDSs' to store the CDSs of interest as fasta files for future processing. In case the corresponding fasta file already exists within the 'set_CDSs' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded CDSs.

Author(s)

Hajk-Georg Drost

See Also

getGenomeSet, getProteomeSet, getRNASet, getGFFSet, getCDS, getGFF, getRNA, meta.retrieval, read_cds

Examples

```
## Not run:
getCDSSet("refseq", organisms = c("Arabidopsis thaliana",
                                  "Arabidopsis lyrata",
                                  "Capsella rubella"))

## End(Not run)
```

---

getCollection *Retrieve a Collection: Genome, Proteome, CDS, RNA, GFF, Repeat Masker, AssemblyStats*

Description

Main collection retrieval function for an organism of interest. By specifying the scientific name of an organism of interest a collection consisting of the genome file, proteome file, CDS file, RNA file, GFF file, Repeat Masker file, AssemblyStats file of the organism of interest can be downloaded and stored locally. Collections can be retrieved from several databases.

Usage

```
getCollection(
  db = "refseq",
  organism,
  reference = TRUE,
  release = NULL,
  gunzip = FALSE,
  remove_annotation_outliers = FALSE,
  path = file.path("_db_downloads", "collections")
)
```

Arguments

db	a character string specifying the database from which the collection shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organism	there are three options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens"

- by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens)
- by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)

reference — a logical value indicating whether or not a collection shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.

release — the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.

gunzip — a logical value indicating whether or not files should be unzipped.

remove_annotation_outliers — shall outlier lines be removed from the input annotation_file? If yes, then the initial annotation_file will be overwritten and the removed outlier lines will be stored at tempdir for further exploration.

path — a character string specifying the location (a folder) in which the corresponding collection shall be stored. Default is path = file.path("_db_downloads","collections").

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory '_ncbi_downloads/collection' to store the genome of interest as fasta file for future processing. In case the corresponding fasta file already exists within the '_ncbi_downloads/collection' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded genome.

Author(s)

Hajk-Georg Drost

See Also

getGenomeSet, getProteomeSet, getCDSSet, getGenome, getProteome, getCDS, getGFF, getRNA, meta.retrieval, read_genome

Examples

```
## Not run:

# download the collection of Arabidopsis thaliana from refseq
# and store the corresponding genome file in '_ncbi_downloads/collection'
 getCollection( db       = "refseq",
             organism = "Arabidopsis thaliana",
             path = file.path("_db_downloads","collections"))
```

```
## End(Not run)
```

`getCollectionSet` *Retrieve a Collection: Genome, Proteome, CDS, RNA, GFF, Repeat Masker, AssemblyStats of multiple species*

Description

Main collection retrieval function for an organism of interest. By specifying the scientific name of an organism of interest a collection consisting of the genome file, proteome file, CDS file, RNA file, GFF file, Repeat Masker file, AssemblyStats file of the organism of interest can be downloaded and stored locally. Collections can be retrieved from several databases.

Usage

```
getCollectionSet(
  db = "refseq",
  organisms,
  reference = FALSE,
  release = NULL,
  clean_retrieval = FALSE,
  gunzip = TRUE,
  update = FALSE,
  remove_annotation_outliers = TRUE,
  path = "set_collections"
)
```

Arguments

db	a character string specifying the database from which the collection shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organisms	a character vector storing the scientific names of the organisms for which collections shall be retrieved. There are three options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a collection shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.

release — the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.

clean_retrieval, — a logical, default FALSE. Cleaning file names for more convenient downstream processing.

gunzip — a logical value indicating whether or not files should be unzipped.

update — a logical, default FALSE. The existing file will be retained if existing. If TRUE, will download and overwrite the file.

remove_annotation_outliers — shall outlier lines be removed from the input annotation_file? If yes, then the initial annotation_file will be overwritten and the removed outlier lines will be stored at tempdir for further exploration.

path — a character string specifying the location (a folder) in which the corresponding collection shall be stored. Default is path = file.path("_db_downloads","collections").

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory '_ncbi_downloads/collection' to store the genome of interest as fasta file for future processing. In case the corresponding fasta file already exists within the '_ncbi_downloads/collection' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded genome.

Author(s)

Hajk-Georg Drost

See Also

getCollection, getGenomeSet, getProteomeSet, getCDSSet, getGenome, getProteome, getCDS, getGFF, getRNA, meta.retrieval, read_genome

Examples

```
## Not run:
# define scientific names of species for which
# collections shall be retrieved
organism_list <- c("Arabidopsis thaliana",
                   "Arabidopsis lyrata",
                   "Capsella rubella")
# download the collection of Arabidopsis thaliana from refseq
# and store the corresponding genome file in '_ncbi_downloads/collection'
 getCollectionSet( db       = "refseq",
```

```
              organism = organism_list,
              path = "set_collections")

## End(Not run)
```

---

getDatasets *Retrieve All Available Datasets for a BioMart Database*

Description

This funcion queries the BioMart API and returns a table storing all available datasets for a selected BioMart databases.

Usage

```
getDatasets(mart)
```

Arguments

mart	a character string specifying the database (mart) for which datasets shall be listed.

Author(s)

Hajk-Georg Drost

See Also

getMarts, getAttributes, getFilters, organismBM, organismFilters, organismAttributes

Examples

```
## Not run:
# search for available datasets
# getMarts()
# choose database: "ENSEMBL_MART_ENSEMBL"
head(getDatasets("ENSEMBL_MART_ENSEMBL"), 10)

## End(Not run)
```

`getENSEMBL.gtf` *Helper function for retrieving gff files from ENSEMBL*

Description

This function downloads gff files of query organisms from ENSEMBL.

Usage

```
getENSEMBL.gtf(
  organism,
  type = "dna",
  id.type = "toplevel",
  path,
  release = NULL
)
```

Arguments

`organism`	scientific name of the organism of interest.
`type`	biological sequence type.
`id.type`	a character, default "toplevel". id type of assembly, either toplevel or primary_assembly usually.
`path`	location where file shall be stored.
`release`	a numeric, the database release version of ENSEMBL (`db = "ensembl"`). Default is `release = NULL` meaning that the most recent database version is used. `release = 75` would for human would give the stable GRCh37 release in ensembl. Value must be > 46, since ensembl did not structure their data if the standard format before that.

Value

character filepath to download file, returns FALSE if failed.

Author(s)

Hajk-Georg Drost

`getENSEMBL.Seq` *Helper function for retrieving biological sequence files from ENSEMBL*

Description

This function downloads gff files of query organisms from ENSEMBL.

Usage

```
getENSEMBL.Seq(
  organism,
  type = "dna",
  id.type = "toplevel",
  release = NULL,
  path
)
```

Arguments

`organism`	scientific name of the organism of interest.
`type`	biological sequence type.
`id.type`	a character, default "toplevel". id type of assembly, either toplevel or primary_assembly usually.
`release`	a numeric, the database release version of ENSEMBL (`db = "ensembl"`). Default is `release = NULL` meaning that the most recent database version is used. `release = 75` would for human would give the stable GRCh37 release in ensembl. Value must be > 46, since ensembl did not structure their data if the standard format before that.
`path`	location where file shall be stored.

Value

either a character path to downloaded file, or a logical FALSE, specifying failure.

Author(s)

Hajk-Georg Drost

getENSEMBLGENOMESInfo *Retrieve ENSEMBLGENOMES info file*

Description

Retrieve species and genome information from http://rest.ensemblgenomes.org/info/species?content-type=application/json/.

Usage

```
getENSEMBLGENOMESInfo()
```

Author(s)

Hajk-Georg Drost

Examples

```
## Not run:
info.file <- getENSEMBLGENOMESInfo()
info.file

## End(Not run)
```

getENSEMBLInfo *Retrieve ENSEMBL info file*

Description

Retrieve species and genome information from http://rest.ensembl.org/info/species?content-type=application/json/.

Usage

```
getENSEMBLInfo()
```

Author(s)

Hajk-Georg Drost

getFilters *Retrieve All Available Filters for a Specific Dataset*

Description

This funcion queries the BioMart API and returns a table storing all available filters for a specific dataset.

Usage

```
getFilters(mart, dataset)
```

Arguments

mart	a character string specifying the database (mart) for which datasets shall be listed.
dataset	a character string specifying the dataset for which filters shall be listed.

Author(s)

Hajk-Georg Drost

See Also

getMarts, getDatasets, getAttributes, organismBM, organismFilters, organismAttributes

Examples

```
## Not run:
# search for available datasets
# getMarts()
# choose database (mart): "ENSEMBL_MART_ENSEMBL"
# head(getDatasets(mart = "ENSEMBL_MART_ENSEMBL"), 10)
# choose dataset: "hsapiens_gene_ensembl"
head(getFilters(mart = "ENSEMBL_MART_ENSEMBL",
                dataset = "hsapiens_gene_ensembl") , 5)

## End(Not run)
```

getGenome *Genome Retrieval*

Description

Main genome retrieval function for an organism of interest. By specifying the scientific name of an organism of interest the corresponding fasta-file storing the genome of the organism of interest can be downloaded and stored locally. Genome files can be retrieved from several databases. In addition, the genome summary statistics for the retrieved species is stored locally to provide users with insights regarding the genome assembly quality (see summary_genome for details). This is useful when comparing genomes with large difference in genome assembly qualities.

Usage

```
getGenome(
  db = "refseq",
  organism,
  reference = FALSE,
  release = NULL,
  gunzip = FALSE,
  path = file.path("_ncbi_downloads", "genomes"),
  assembly_type = "toplevel"
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organism	there are three options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
release	a numeric, the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used. release = 75 would for human would give the stable GRCh37 release in ensembl. Value must be > 46, since ensembl did not structure their data if the standard format before that.
gunzip	a logical value indicating whether or not files should be unzipped.

path — a character string specifying the location (a folder) in which the corresponding genome shall be stored. Default is `path = file.path("_ncbi_downloads","genomes")`.

assembly_type — a character string specifying from which assembly type the genome shall be retrieved from (ensembl only, else this argument is ignored): Default is `assembly_type = "toplevel"`). This will give you all multi-chromosomes (copies of the same chromosome with small variations). As an example the toplevel fasta genome in human is over 70 GB uncompressed. To get primary assembly with 1 chromosome variant per chromosome: `assembly_type = "primary_assembly"`). As an example, the primary_assembly fasta genome in human is only a few GB uncompressed:

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory '_ncbi_downloads/genomes' to store the genome of interest as fasta file for future processing. In case the corresponding fasta file already exists within the '_ncbi_downloads/genomes' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded genome.

Author(s)

Hajk-Georg Drost

See Also

getGenomeSet, getProteome, getCDS, getGFF, getRNA, getRepeatMasker, getAssemblyStats, summary_genome, meta.retrieval, meta.retrieval.all, read_genome

Examples

```
## Not run:

# download the genome of Arabidopsis thaliana from refseq
# and store the corresponding genome file in '_ncbi_downloads/genomes'
file_path <- getGenome( db       = "refseq",
             organism = "Arabidopsis thaliana",
             path = file.path("_ncbi_downloads","genomes"))

Ath_genome <- read_genome(file_path, format = "fasta")


# download the genome of Arabidopsis thaliana from genbank
# and store the corresponding genome file in '_ncbi_downloads/genomes'
file_path <- getGenome( db       = "genbank",
             organism = "Arabidopsis thaliana",
```

```
                path = file.path("_ncbi_downloads","genomes"))

Ath_genome <- read_genome(file_path, format = "fasta")

## End(Not run)
```

---

getGENOMEREPORT *Retrieve NCBI GENOME_REPORTS file*

Description

Retrieves NCBI GENOME_REPORTS file from ftp://ftp.ncbi.nlm.nih.gov/genomes/GENOME_REPORTS/overview.txt.

Usage

```
getGENOMEREPORT()
```

Author(s)

Hajk-Georg Drost

Examples

```
## Not run:
report <- getGENOMEREPORT()
report

## End(Not run)
```

---

getGenomeSet *Genome Retrieval of multiple species*

Description

Main genome retrieval function for a set of organism of interest. By specifying the scientific names of the organisms of interest the corresponding fasta-files storing the genome of the organisms of interest will be downloaded and stored locally. Genome files can be retrieved from several databases.

Usage

```
getGenomeSet(
  db = "refseq",
  organisms,
  reference = FALSE,
  release = NULL,
  clean_retrieval = TRUE,
  gunzip = TRUE,
  update = FALSE,
  path = "set_genomes",
  assembly_type = "toplevel"
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organisms	a character vector storing the names of the organisms than shall be retrieved. There are three available options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
clean_retrieval	logical value indicating whether or not downloaded files shall be renamed for more convenient downstream data analysis.
gunzip	a logical value indicating whether or not files should be unzipped.
update	a logical value indicating whether or not files that were already downloaded and are still present in the output folder shall be updated and re-loaded (update = TRUE or whether the existing file shall be retained update = FALSE (Default)).
path	a character string specifying the location (a folder) in which the corresponding genomes shall be stored. Default is path = "set_genomes".
assembly_type	a character string specifying from which assembly type the genome shall be retrieved from (ensembl only, else this argument is ignored): Default is assembly_type = "toplevel"). This will give you all multi-chromosomes (copies of the same chromosome with small variations). As an example the toplevel fasta genome in

human is over 70 GB uncompressed. To get primary assembly with 1 chromosome variant per chromosome: `assembly_type = "primary_assembly"`). As an example, the primary_assembly fasta genome in human is only a few GB uncompressed:

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory 'set_genomes' to store the genomes of interest as fasta files for future processing. In case the corresponding fasta file already exists within the 'set_genomes' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded genomes.

Author(s)

Hajk-Georg Drost

See Also

`getProteomeSet`, `getCDSSet`, `getRNASet`, `getGFFSet`, `getCDS`, `getGFF`, `getGTF`, `getRNA`, `meta.retrieval`, `read_genome`

Examples

```
## Not run:
getGenomeSet("refseq", organisms = c("Arabidopsis thaliana",
                                     "Arabidopsis lyrata",
                                     "Capsella rubella"))

## End(Not run)
```

---

getGFF *Genome Annotation Retrieval (GFF3)*

Description

Main retrieval function for GFF files of an organism of interest. By specifying the scientific name of an organism of interest the corresponding gff file storing the annotation for the organism of interest can be downloaded and stored locally. GFF files can be retrieved from several databases.

Usage

```
getGFF(
  db = "refseq",
  organism,
  reference = FALSE,
  release = NULL,
  gunzip = FALSE,
  remove_annotation_outliers = FALSE,
  path = file.path("_ncbi_downloads", "annotation")
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organism	a character string specifying the scientific name of the organism of interest, e.g. organism = "Homo sapiens".
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
gunzip	a logical value indicating whether or not files should be unzipped.
remove_annotation_outliers	shall outlier lines be removed from the input annotation_file? If yes, then the initial annotation_file will be overwritten and the removed outlier lines will be stored at tempdir for further exploration.
path	a character string specifying the location (a folder) in which the corresponding annotation file shall be stored. Default is path = file.path("_ncbi_downloads","genomes").

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory '_ncbi_downloads/annotation' to store the genome of interest as fasta file for future processing. In case the corresponding fasta file already exists within the '_ncbi_downloads/annotation' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded annotation file.

Author(s)

Hajk-Georg Drost

See Also

getProteome, getCDS, getGenome, getRNA, getRepeatMasker, getAssemblyStats, meta.retrieval

Examples

```
## Not run:
# download the annotation of Arabidopsis thaliana from refseq
# and store the corresponding genome file in '_ncbi_downloads/annotation'
getGFF( db       = "refseq",
               organism = "Arabidopsis thaliana",
               path = file.path("_ncbi_downloads","annotation"))


# download the genome of Arabidopsis thaliana from genbank
# and store the corresponding genome file in '_ncbi_downloads/annotation'
getGFF( db       = "genbank",
               organism = "Arabidopsis thaliana",
               path = file.path("_ncbi_downloads","annotation"))


## End(Not run)
```

---

getGFFSet *GFF retrieval of multiple species*

Description

Main GFF retrieval function for a set of organism of interest. By specifying the scientific names of the organisms of interest the corresponding fasta-files storing the GFF of the organisms of interest will be downloaded and stored locally. GFF files can be retrieved from several databases.

Usage

```
getGFFSet(
  db = "refseq",
  organisms,
  reference = FALSE,
  release = NULL,
  clean_retrieval = TRUE,
  gunzip = TRUE,
  remove_annotation_outliers = FALSE,
  update = FALSE,
  path = "set_GFF"
)
```

Arguments

db	a character string specifying the database from which the GFF shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organisms	a character vector storing the names of the organisms than shall be retrieved. There are three available options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a GFF shall be downloaded if it isn't marked in the database as either a reference GFF or a representative GFF
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
clean_retrieval	logical value indicating whether or not downloaded files shall be renamed for more convenient downstream data analysis.
gunzip	a logical value indicating whether or not files should be unzipped.
remove_annotation_outliers	shall outlier lines be removed from the input annotation_file? If yes, then the initial annotation_file will be overwritten and the removed outlier lines will be stored at tempdir for further exploration.
update	a logical value indicating whether or not files that were already downloaded and are still present in the output folder shall be updated and re-loaded (update = TRUE or whether the existing file shall be retained update = FALSE (Default)).
path	a character string specifying the location (a folder) in which the corresponding CDSs shall be stored. Default is path = "set_CDS".

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory 'set_CDSs' to store the CDSs of interest as fasta files for future processing. In case the corresponding fasta file already exists within the 'set_CDSs' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded CDSs.

Author(s)

Hajk-Georg Drost

See Also

getGenomeSet, getProteomeSet, getCDSSet, getRNASet, getGFF, getRNA, meta.retrieval, read_cds

Examples

```
## Not run:
getGFFSet("refseq", organisms = c("Arabidopsis thaliana",
                                  "Arabidopsis lyrata",
                                  "Capsella rubella"))

## End(Not run)
```

---

getGO *Gene Ontology Query*

Description

This function takes a gene id as character vector from a given query organism and returns the corresponding GO terms and additional GO information.

Usage

```
getGO(organism, genes, filters, ...)
```

Arguments

organism	a character string specifying the scientific name of a query organism.
genes	a character vector storing the gene ids of a organisms of interest to be queried against Ensembl Biomart.
filters	a character vector specifying the filter (query key) for the Ensembl Biomart query, e.g. filter = "ensembl_gene_id".
...	additional parameters that can be passed to the biomart function.

Details

This function takes the scientific name of a query organism, a set of genes for which GO terms and additional information shall be retrieved, and a filter argument that specifies the attribute for the query genes.

Author(s)

Hajk-Georg Drost

See Also

biomart, organismFilters, organismBM, getBM, getMarts, getDatasets, getFilters

Examples

```
## Not run:
GO_tbl <- getGO(organism = "Arabidopsis thaliana",
                genes    = c("AT1G06090", "AT1G06100"),
                filters  = "ensembl_gene_id")

# look at the result
head(GO_tbl)

## End(Not run)
```

---

getGroups *Retrieve available groups for a kingdom of life (only available for NCBI RefSeq and NCBI Genbank)*

Description

A short list of available groups for a kingdom of life.

Usage

```
getGroups(db = "refseq", kingdom)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" Default is db = "refseq".
kingdom	a character string specifying for which kingdom of life groups shall be retrieved. See getKingdoms for details.

Author(s)

Hajk-Georg Drost

See Also

meta.retrieval, getGenome, getProteome, getCDS, getKingdoms

Examples

```
# get possible kigdom names
getKingdoms(db = "refseq")
## Not run:
# retrieve subgroups for vertebrate_mammalian available from refseq
getGroups(db = "refseq", kingdom = "vertebrate_mammalian")

# get possible kigdom names
getKingdoms(db = "genbank")
# retrieve subgroups for vertebrate_mammalian available from genbank
getGroups(db = "genbank", kingdom = "vertebrate_mammalian")

## End(Not run)
```

---

getGTF *Genome Annotation Retrieval (GTF)*

Description

Main retrieval function for GTF files of an organism of interest. By specifying the scientific name of an organism of interest the corresponding GTF file storing the annotation for the organism of interest can be downloaded and stored locally. GTF files can be retrieved from several databases.

Usage

```
getGTF(
  db = "ensembl",
  organism,
  remove_annotation_outliers = FALSE,
  path = file.path("ensembl", "annotation"),
  assembly_type = "toplevel",
  release = NULL
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "ensembl"
organism	a character string specifying the scientific name of the organism of interest, e.g. organism = "Homo sapiens".
remove_annotation_outliers	shall outlier lines be removed from the input annotation_file? If yes, then the initial annotation_file will be overwritten and the removed outlier lines will be stored at tempdir for further exploration.
path	a character string specifying the location (a folder) in which the corresponding annotation file shall be stored. Default is path = file.path("ensembl","annotation").

assembly_type — a character string specifying from which assembly type the genome shall be retrieved from (ensembl only, else this argument is ignored): Default is `assembly_type = "toplevel"`). This will give you all multi-chromosomes (copies of the same chromosome with small variations). As an example the toplevel fasta genome in human is over 70 GB uncompressed. To get primary assembly with 1 chromosome variant per chromosome: `assembly_type = "primary_assembly"`). As an example, the primary_assembly fasta genome in human is only a few GB uncompressed:

release — a numeric, the database release version of ENSEMBL (`db = "ensembl"`). Default is `release = NULL` meaning that the most recent database version is used. `release = 75` would for human would give the stable GRCh37 release in ensembl. Value must be > 46, since ensembl did not structure their data if the standard format before that.

Details

Internally this function loads the the overview.txt file from ENSEMBL: and creates a directory 'ensembl/annotation' to store the genome of interest as fasta file for future processing. In case the corresponding fasta file already exists within the 'ensembl/annotation' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded annotation file.

Author(s)

Hajk-Georg Drost

See Also

getProteome, getCDS, getGenome, getRNA, getRepeatMasker, getAssemblyStats, meta.retrieval, getGFF

Examples

```
## Not run:
# download the annotation of Homo sapiens from ensembl
# and store the corresponding genome file in 'ensembl/annotation'
getGTF(db            = "ensembl",
       organism      = "Homo sapiens",
       path          = file.path("ensembl","annotation"))

getGTF(db            = "ensembl",
       organism      = "Homo sapiens",
       path          = file.path("ensembl","annotation"),
       assembly_type = "primary_assembly")


## End(Not run)
```

getKingdomAssemblySummary *Retrieve and summarise the assembly_summary.txt files from NCBI for all kingdoms*

Description

Retrieval function of the assembly_summary.txt file from NCBI for all kingdoms. The assembly_summary.txt files store available species on NCBI.

Usage

```
getKingdomAssemblySummary(db)
```

Arguments

db	database name. E.g. `refseq` or genbank.

Author(s)

Hajk-Georg Drost

See Also

`getSummaryFile`, `getMetaGenomeSummary`

Examples

```
## Not run:
test <- getKingdomAssemblySummary(db = "refseq")
test

## End(Not run)
```

getKingdoms *Retrieve available kingdoms of life*

Description

A short list of available kingdoms of life

Usage

```
getKingdoms(db = "refseq")
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: db = "refseq", db = "genbank", db = "ensembl", db = "ensemblgenomes". Default is db = "refseq".

Author(s)

Hajk-Georg Drost

See Also

meta.retrieval, getGenome, getProteome, getCDS, getGroups

Examples

```
# retrieve kingdoms available from refseq
getKingdoms(db = "refseq")

# retrieve kingdoms available from genbank
getKingdoms(db = "genbank")
```

---

getMarts *Retrieve information about available Ensembl Biomart databases*

Description

This funcion queries the Ensembl Biomart API and returns a table storing information about all available Ensembl Biomart databases.

Usage

```
getMarts()
```

Author(s)

Hajk-Georg Drost

See Also

getDatasets, getAttributes, getFilters, organismBM, organismFilters, organismAttributes

Examples

```
## Not run:
# get a table of all available databases from Ensembl Biomart
getMarts()

## End(Not run)
```

getMetaGenomeAnnotations *Retrieve annotation *.gff files for metagenomes from NCBI Genbank*

Description

Retrieve available annotation *.gff files for metagenomes from NCBI Genbank. NCBI Genbank allows users to download entire metagenomes and their annotations of several metagenome projects. This function downloads available metagenomes that can then be downloaded via getMetaGenomes.

Usage

```
getMetaGenomeAnnotations(
  name,
  path = file.path("_ncbi_downloads", "metagenome", "annotations")
)
```

Arguments

name	metagenome name retrieved by listMetaGenomes.
path	a character string specifying the location (a folder) in which the corresponding metagenome annotations shall be stored. Default is path = file.path("_ncbi_downloads","metageno

Author(s)

Hajk-Georg Drost

See Also

getMetaGenomes, listMetaGenomes, getGFF

Examples

```
## Not run:
# Frist, retrieve a list of available metagenomes
listMetaGenomes()

# Now, retrieve the 'human gut metagenome'
getMetaGenomeAnnotations(name = "human gut metagenome")

## End(Not run)
```

getMetaGenomes *Retrieve metagenomes from NCBI Genbank*

Description

Retrieve available metagenomes from NCBI Genbank. NCBI Genbank allows users to download entire metagenomes of several metagenome projects. This function downloads available metagenomes that can then be downloaded via getMetaGenomes.

Usage

```
getMetaGenomes(name, path = file.path("_ncbi_downloads", "metagenome"))
```

Arguments

name	metagenome name retrieved by listMetaGenomes.
path	a character string specifying the location (a folder) in which the corresponding metagenome shall be stored. Default is path = file.path("_ncbi_downloads","metagenome").

Author(s)

Hajk-Georg Drost

See Also

getMetaGenomeAnnotations, listMetaGenomes

Examples

```
## Not run:
# Frist, retrieve a list of available metagenomes
listMetaGenomes()

# Now, retrieve the 'human gut metagenome'
getMetaGenomes(name = "human gut metagenome")

## End(Not run)
```

getMetaGenomeSummary *Retrieve the assembly_summary.txt file from NCBI genbank metagenomes*

Description

Retrieval function of the assembly_summary.txt file from NCBI genbank metagenomes. This files stores all available metagenome projects on NCBI Genbank.

Usage

```
getMetaGenomeSummary()
```

Author(s)

Hajk-Georg Drost

See Also

getKingdomAssemblySummary, getSummaryFile

Examples

```
## Not run:
meta.summary <- getMetaGenomeSummary()
meta.summary

## End(Not run)
```

getProteome *Proteome Retrieval*

Description

Main proteome retrieval function for an organism of interest. By specifying the scientific name of an organism of interest the corresponding fasta-file storing the proteome of the organism of interest can be downloaded and stored locally. Proteome files can be retrieved from several databases.

Usage

```
getProteome(
  db = "refseq",
  organism,
  reference = TRUE,
  release = NULL,
  gunzip = FALSE,
  path = file.path("_ncbi_downloads", "proteomes")
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl" • db = "uniprot"
organism	there are three options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
gunzip	a logical value indicating whether or not files should be unzipped.
path	a character string specifying the location (a folder) in which the corresponding proteome shall be stored. Default is path = file.path("_ncbi_downloads","proteomes").

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory '_ncbi_downloads/proteomes' to store the proteome of interest as fasta file for future processing.

Value

File path to downloaded proteome.

Author(s)

Hajk-Georg Drost

See Also

getGenome, getCDS, getGFF, getRNA, getRepeatMasker, getAssemblyStats, meta.retrieval, read_proteome

Examples

```
## Not run:

# download the proteome of Arabidopsis thaliana from refseq
# and store the corresponding proteome file in '_ncbi_downloads/proteomes'
file_path <- getProteome( db       = "refseq",
             organism = "Arabidopsis thaliana",
             path     = file.path("_ncbi_downloads","proteomes") )

Ath_proteome <- read_proteome(file_path, format = "fasta")

# download the proteome of Arabidopsis thaliana from genbank
# and store the corresponding proteome file in '_ncbi_downloads/proteomes'
file_path <- getProteome( db       = "genbank",
             organism = "Arabidopsis thaliana",
             path     = file.path("_ncbi_downloads","proteomes") )

Ath_proteome <- read_proteome(file_path, format = "fasta")

## End(Not run)
```

`getProteomeSet` *Proteome retrieval of multiple species*

Description

Main proteome retrieval function for a set of organism of interest. By specifying the scientific names of the organisms of interest the corresponding fasta-files storing the proteome of the organisms of interest will be downloaded and stored locally. proteome files can be retrieved from several databases.

Usage

```
getProteomeSet(
  db = "refseq",
  organisms,
  reference = FALSE,
  release = NULL,
  clean_retrieval = TRUE,
  gunzip = TRUE,
  update = FALSE,
  path = "set_proteomes"
)
```

Arguments

db	a character string specifying the database from which the proteome shall be retrieved:

- db = "refseq"
- db = "genbank"
- db = "ensembl"

organisms — a character vector storing the names of the organisms than shall be retrieved. There are three available options to characterize an organism:

- by scientific name: e.g. organism = "Homo sapiens"
- by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens)
- by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)

reference — a logical value indicating whether or not a proteome shall be downloaded if it isn't marked in the database as either a reference proteome or a representative proteome.

release — the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.

clean_retrieval — logical value indicating whether or not downloaded files shall be renamed for more convenient downstream data analysis.

gunzip — a logical value indicating whether or not files should be unzipped.

update — a logical value indicating whether or not files that were already downloaded and are still present in the output folder shall be updated and re-loaded (update = TRUE or whether the existing file shall be retained update = FALSE (Default)).

path — a character string specifying the location (a folder) in which the corresponding proteomes shall be stored. Default is path = "set_proteomes".

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory 'set_proteomes' to store the proteomes of interest as fasta files for future processing. In case the corresponding fasta file already exists within the 'set_proteomes' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded proteomes.

Author(s)

Hajk-Georg Drost

See Also

getGenomeSet, getCDSSet, getRNASet, getGFFSet, getCDS, getGFF, getRNA, meta.retrieval, read_proteome

Examples

```
## Not run:
getProteomeSet("refseq", organisms = c("Arabidopsis thaliana",
                                       "Arabidopsis lyrata",
                                        "Capsella rubella"))

## End(Not run)
```

`getReleases` *Retrieve available database releases or versions of ENSEMBL*

Description

Retrieve available database releases or versions of ENSEMBL.

Usage

```
getReleases(db = "ensembl")
```

Arguments

db	a character string specifying the database from which available resease versions shall be retrieved: • db = "ensembl"

Author(s)

Hajk-Georg Drost

Examples

```
## Not run:
# retrieve available resease versions of ENSEMBL
getReleases("ensembl")

## End(Not run)
```

`getRepeatMasker` *Repeat Masker Retrieval*

Description

Main Repeat Masker output retrieval function for an organism of interest. By specifying the scientific name of an organism of interest the corresponding Repeat Masker file storing the genome of the organism of interest can be downloaded and stored locally. Repeat Masker files can be retrieved from several databases.

Usage

```
getRepeatMasker(
  db = "refseq",
  organism,
  reference = FALSE,
  path = file.path("_ncbi_downloads", "repeatmasker")
)
```

Arguments

`db`	a character string specifying the database from which the genome shall be retrieved: • `db = "refseq"` • `db = "genbank"`
`organism`	a character string specifying the scientific name of the organism of interest, e.g. `organism = "Homo sapiens"`.
`reference`	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
`path`	a character string specifying the location (a folder) in which the corresponding file shall be stored. Default is `path = file.path("_ncbi_downloads","repeatmasker")`.

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory '_ncbi_downloads/repeatmasker' to store the files of interest as fasta file for future processing. In case the corresponding fasta file already exists within the '_ncbi_downloads/repeatmasker' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded Repeat Masker output file.

Author(s)

Hajk-Georg Drost

See Also

getProteome, getCDS, getGFF, getRNA, meta.retrieval, read_rm, getGenome

Examples

```
## Not run:

# download the Repeat Masker output file of Arabidopsis thaliana from refseq
# and store the corresponding genome file in '_ncbi_downloads/genomes'
file_path <- getRepeatMasker( db       = "refseq",
             organism = "Arabidopsis thaliana",
             path = file.path("_ncbi_downloads","repeatmasker"))

Ath_repeatmasker <- read_rm(file_path)


# download the Repeat Masker output file of Arabidopsis thaliana from genbank
# and store the corresponding genome file in '_ncbi_downloads/genomes'
file_path <- getRepeatMasker( db       = "genbank",
             organism = "Arabidopsis thaliana",
             path = file.path("_ncbi_downloads","repeatmasker"))

Ath_repeatmasker <- read_rm(file_path)

## End(Not run)
```

---

getRNA *RNA Sequence Retrieval*

Description

Main retrieval function for RNA sequences of an organism of interest. By specifying the scientific name of an organism of interest the corresponding fasta-file storing the RNA information for the organism of interest can be downloaded and stored locally. RNA files can be retrieved from several databases.

Usage

```
getRNA(
  db = "refseq",
  organism,
  reference = FALSE,
  release = NULL,
  path = file.path("_ncbi_downloads", "RNA")
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organism	there are three options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome.
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
path	a character string specifying the location (a folder) in which the corresponding CDS file shall be stored. Default is path = file.path("_ncbi_downloads","RNA").

Value

File path to downloaded RNA file.

Author(s)

Hajk-Georg Drost

See Also

getGenome, getProteome, getGTF, getGFF, getRepeatMasker, getAssemblyStats, meta.retrieval, read_cds, getCDS

Examples

```
## Not run:
# download the RNA of Arabidopsis thaliana from refseq
# and store the corresponding RNA file in '_ncbi_downloads/RNA'
file_path <- getRNA( db       = "refseq",
             organism = "Arabidopsis thaliana",
             path     = file.path("_ncbi_downloads","RNA"))

Ath_RNA <- read_rna(file_path, format = "fasta")

## End(Not run)
```

getRNASet *RNA Retrieval of multiple species*

Description

Main RNA retrieval function for a set of organism of interest. By specifying the scientific names of the organisms of interest the corresponding fasta-files storing the RNA of the organisms of interest will be downloaded and stored locally. RNA files can be retrieved from several databases.

Usage

```
getRNASet(
  db = "refseq",
  organisms,
  reference = FALSE,
  release = NULL,
  clean_retrieval = TRUE,
  gunzip = TRUE,
  update = FALSE,
  path = "set_RNAs"
)
```

Arguments

db	a character string specifying the database from which the RNA shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl"
organisms	a character vector storing the names of the organisms than shall be retrieved. There are three available options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
reference	a logical value indicating whether or not a RNA shall be downloaded if it isn't marked in the database as either a reference RNA or a representative RNA
release	the database release version of ENSEMBL (db = "ensembl"). Default is release = NULL meaning that the most recent database version is used.
clean_retrieval	logical value indicating whether or not downloaded files shall be renamed for more convenient downstream data analysis.
gunzip	a logical value indicating whether or not files should be unzipped.

update	a logical value indicating whether or not files that were already downloaded and are still present in the output folder shall be updated and re-loaded (update = TRUE or whether the existing file shall be retained update = FALSE (Default)).
path	a character string specifying the location (a folder) in which the corresponding RNAs shall be stored. Default is path = "set_RNAs".

Details

Internally this function loads the the overview.txt file from NCBI:

refseq: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/

genbank: ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/

and creates a directory 'set_RNAs' to store the RNAs of interest as fasta files for future processing. In case the corresponding fasta file already exists within the 'set_RNAs' folder and is accessible within the workspace, no download process will be performed.

Value

File path to downloaded RNAs.

Author(s)

Hajk-Georg Drost

See Also

getGenomeSet, getRNASet, getProteomeSet, getGFFSet, getCDS, getGFF, getRNA, meta.retrieval, read_rna

Examples

```
## Not run:
getRNASet("refseq", organisms = c("Arabidopsis thaliana",
                                  "Arabidopsis lyrata",
                                  "Capsella rubella"))

## End(Not run)
```

---

getSummaryFile *Helper function to retrieve the assembly_summary.txt file from NCBI*

Description

Retrieval function of the assembly_summary.txt file from NCBI.

Usage

```
getSummaryFile(db, kingdom)
```

Arguments

db	database name. E.g. refseq or genbank.
kingdom	kingdom for which assembly_summary.txt file shall be retrieved. See also getKingdoms.

Author(s)

Hajk-Georg Drost

See Also

getKingdomAssemblySummary, getMetaGenomeSummary

Examples

```
## Not run:
test <- getSummaryFile("refseq","plant")
test

## End(Not run)
```

---

is.genome.available *Check Genome Availability*

Description

This function checks the availability of a given genome on the NBCI servers specified as scientific name.

Usage

```
is.genome.available(db = "refseq", organism, details = FALSE)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "ensembl" • db = "uniprot"
organism	there are three options to characterize an organism: • by scientific name: e.g. organism = "Homo sapiens" • by database specific accession identifier: e.g. organism = "GCF_000001405.37" (= NCBI RefSeq identifier for Homo sapiens) • by taxonomic identifier from NCBI Taxonomy: e.g. organism = "9606" (= taxid of Homo sapiens)
details	a logical value specifying whether or not details on genome size, kingdom, etc. shall be printed to the console intead of a boolean value.

Details

Internally this function calls the `listGenomes` function to detect all available genomes and checks whether or not the specified organism is available for download.

Value

a logical value specifing whether or not the genome of the input organism is available. In case `details = TRUE` only a character string specifying the genome details is being returned.

Author(s)

Hajk-Georg Drost

Examples

```
## Not run:
# checking whether the Homo sapiens genome is stored on NCBI
is.genome.available(organism = "Homo sapiens", db = "refseq")

# and printing details
is.genome.available(organism = "Homo sapiens", db = "refseq", details = TRUE)

# checking whether the Homo sapiens genome is stored on ENSEMBL
is.genome.available(organism = "Homo sapiens", db = "ensembl")

# and printing details
is.genome.available(organism = "Homo sapiens",
                    details = TRUE,
                    db = "ensembl")

## End(Not run)
```

`listDatabases` *Retrieve a List of Available NCBI Databases for Download*

Description

This function allows you to retrieve a list of database names and versions that can be downloaded from correspondning servers.

Database retrieval is crucial for most biological studies and analyses. There is a vast diversity of databases that can be accessed remotely or that can be downloaded to your local machine. This function provides an interface to databases that can be downloaded from NCBI servers and lists all available databases and their database version to be able to select an appropriate database for download with `download.database`.

Usage

```
listDatabases(db = "nr", update = FALSE)

listNCBIDatabases(db = "nr", update = FALSE)
```

Arguments

db	a character string specifying the name of the database that shall be searched for.
update	a logical value specifying whether or not the local listDatabases.txt file shall be updated by remote access to NCBI.

Author(s)

Hajk-Georg Drost

See Also

download.database, download.database.all

Examples

```
## Not run:
# retrieve all versions of the NCBI 'nr' database that can be downloaded
listNCBIDatabases(db = "nr")

# analogous:
# listNCBIDatabases(db = "cdd")
# listNCBIDatabases(db = "nt")
# listNCBIDatabases(db = "gss")
# listNCBIDatabases(db = "refseq_protein")

## End(Not run)
```

---

listGenomes *List All Available Genomes either by kingdom, group, or subgroup*

Description

This function retrieves the names of all genomes available on the NCBI ftp:// server and stores the results in a file named 'overview.txt' inside the directory _ncbi_downloads' that is built inside the workspace.

Usage

```
listGenomes(db = "refseq", type = "all", subset = NULL, details = FALSE)
```

Arguments

db	a character string specifying the database for which genome availability shall be checked. Available options are: • db = "refseq" • db = "genbank" • db = "ensembl"
type	a character string specifying a potential filter of available genomes. Available options are: • type = "all" • type = "kingdom" • type = "group" • type = "subgroup"
subset	a character string or character vector specifying a subset of type. E.g. if users are interested in retrieving all Eukaryota species, they can specify: type = "kingdom" and subset = "Eukaryota".
details	a boolean value specifying whether only the scientific names of stored genomes shall be returned (details = FALSE) or all information such as • organism_name • kingdoms • group • subgroup • file_size_MB, etc.

Details

Internally this function loads the the overview.txt file from NCBI and creates a directory '_ncbi_downloads' in the temdir() folder to store the overview.txt file for future processing. In case the overview.txt file already exists within the '_ncbi_downloads' folder and is accessible within the workspace, no download process will be performed again.

Note

Please note that the ftp:// connection relies on the NCBI or ENSEMBL server and cannot be accurately accessed via a proxy.

Author(s)

Hajk-Georg Drost

Examples

```
## Not run:
# print details for refseq
listGenomes(db = "refseq")
# print details for all plants in refseq
listGenomes(db = "refseq", type = "kingdom")
```

```
# print details for all plant groups in refseq
listGenomes(db = "refseq", type = "group")
# print details for all plant subgroups in refseq
listGenomes(db = "refseq", type = "subgroup")

## End(Not run)
```

`listGroups` *List number of available genomes in each taxonomic group*

Description

Users can retrieve the available number of sequenced genomes per group. Only available for db = "refseq" and db = "genbank".

Usage

```
listGroups(db = "refseq", kingdom = "all", details = FALSE)
```

Arguments

db	a character string specifying the database for which genome availability shall be checked. Available options are: • db = "refseq" • db = "genbank"
kingdom	a kingdom specification retrieved by getKingdoms.
details	shall all species corresponding to the specified kingdom be returned? Default is details = FALSE.

Author(s)

Hajk-Georg Drost

See Also

listGenomes, is.genome.available, listKingdoms

Examples

```
## Not run:
# example for refseq
listGroups(db = "refseq")
# example for genbank
listGroups(db = "genbank")
### in case groups should be specified by kingdom
# first, retrieve available kingdom names
listKingdoms()
# now we choose kingdom "bacteria"
```

```
listGroups(db = "refseq", kingdom = "bacteria")
# or
listGroups(db = "genbank", kingdom = "bacteria")

## End(Not run)
```

`listKingdoms` *List number of available genomes in each kingdom of life*

Description

Users can retrieve the available number of sequenced genomes per kingdom.

Usage

```
listKingdoms(db = "refseq")
```

Arguments

db	a character string specifying the database for which genome availability shall be checked, e.g. db = "refseq", db = "genbank", db = "ensembl", db = "ensemblgenomes".

Author(s)

Hajk-Georg Drost

See Also

`listGenomes`, `is.genome.available`, `listGroups`

Examples

```
## Not run:
# list number of available genomes in refseq for each kingdom of life
listKingdoms(db = "refseq")
# example for genbank
listKingdoms(db = "genbank")
# example for ensembl
listKingdoms(db = "ensembl")
# example for ensemblgenomes
listKingdoms(db = "ensemblgenomes")

## End(Not run)
```

`listMetaGenomes` *List available metagenomes on NCBI Genbank*

Description

List available metagenomes on NCBI genbank. NCBI genbank allows users to download entire metagenomes of several metagenome projects. This function lists all available metagenomes that can then be downloaded via `getMetaGenomes`.

Usage

```
listMetaGenomes(details = FALSE)
```

Arguments

`details`	a boolean value specifying whether only the scientific names of stored metagenomes shall be returned (`details = FALSE`) or all information such as "organism_name","bioproject", etc (`details = TRUE`).

Author(s)

Hajk-Georg Drost

See Also

`getMetaGenomes`, `getMetaGenomeSummary`

Examples

```
## Not run:
# retrieve available metagenome projects at NCBI Genbank
listMetaGenomes()
# retrieve detailed information on available metagenome projects
# at NCBI Genbank
listMetaGenomes(details = TRUE)

## End(Not run)
```

`meta.retrieval` *Perform Meta-Genome Retrieval*

Description

Download genomes, proteomes, cds, gff, rna, or assembly stats files of all species within a kingdom of life. After downloading users can unzip all files using `clean.retrieval`.

Usage

```
meta.retrieval(
  db = "refseq",
  kingdom,
  group = NULL,
  type = "genome",
  restart_at_last = TRUE,
  reference = FALSE,
  combine = FALSE,
  path = NULL
)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "emsembl"
kingdom	a character string specifying the kingdom of the organisms of interest, e.g. • For NCBI RefSeq: – kingdom = "archaea" – kingdom = "bacteria" – kingdom = "fungi" – kingdom = "invertebrate" – kingdom = "plant" – kingdom = "protozoa" – kingdom = "viral" – kingdom = "vertebrate_mammalian" – kingdom = "vertebrate_other" • For NCBI Genbank: – kingdom = "archaea" – kingdom = "bacteria" – kingdom = "fungi"

  - kingdom = "invertebrate"
  - kingdom = "plant"
  - kingdom = "protozoa"
  - kingdom = "vertebrate_mammalian"
  - kingdom = "vertebrate_other"
- For ENSEMBL:
  - kingdom = "Ensembl"

Available kingdoms can be retrieved with getKingdoms.

group — only species belonging to this subgroup will be downloaded. Groups can be retrieved with getGroups.

type — type of sequences that shall be retrieved. Options are:

- type = "genome" : (for genome assembly retrieval; see also getGenome),
- type = "proteome" : (for proteome retrieval; see also getProteome),
- type = "cds" : (for coding sequence retrieval; see also getCDS),
- type = "gff" : (for annotation file retrieval in gff format; see also getGFF),
- type = "gtf" : (for annotation file retrieval in gtf format (only for ensembl and ensemblgenomes); see also getGTF)
- type = "rna" : (for RNA file retrieval in fasta format; see also getRNA),
- type = "rm" : (for Repeat Masker output file retrieval; see also getRepeatMasker),
- type = "assemblystats" : (for genome assembly quality stats file retrieval; see also getAssemblyStats).

restart_at_last — a logical value indicating whether or not meta.retrieval should pick up at the last species when re-running the function.

- If restart_at_last = TRUE (Default) then meta.retrieval will skip all organisms that are already present in the folder and will start downloading all remaining species. However, this way meta.wretrieval will not be able to check whether already downloaded organism files are corrupted or not by checking the md5 checksum.
- If restart_at_last = FALSE then meta.retrieval will start from the beginning and crawl through already downloaded organism files and check whether already downloaded organism files are corrupted or not by checking the md5 checksum. After checking existing files the function will start downloading all remaining organisms.

reference — a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome. Options are:

- reference = FALSE (Default): all organisms (reference, representative, and non-representative genomes) are downloaded.
- reference = TRUE: organisms that are downloaded must be either a reference or representative genome. Thus, most genomes which are usually non-reference genomes will not be downloaded.

combine — just in case type = "assemblystats" is specified, shall assemby stats of individual species be imported and combined to a data.frame?

path — path to the folder in which downloaded genomes shall be stored. By default the kingdom name is used to name the output folder.

Details

This function aims to perform bulk retrieval of the genomes, proteomes, cds, etc. of species that belong to the same kingdom of life or to the same subgroup.

Value

a character vector storing the file paths of the retrieved files.

Author(s)

Hajk-Georg Drost

See Also

meta.retrieval.all, getCollection, clean.retrieval

Examples

```
## Not run:
# get all available kingdoms for refseq
getKingdoms(db = "refseq")
# download all vertebrate genomes from refseq
meta.retrieval(kingdom = "vertebrate_mammalian",
               db = "refseq",
               type = "genome")

# get all available kingdoms for genbank
getKingdoms(db = "genbank")
# download all vertebrate genomes from genbank
meta.retrieval(kingdom = "vertebrate_mammalian",
               db = "genbank",
               type = "genome")


# In case users do not wish to retrieve genomes from an entire kingdom,
# but rather from a subgoup (e.g. from species belonging to the
# Gammaproteobacteria class, a subgroup of the bacteria kingdom),
# they can use the following workflow"
# First, users can again consult the getKingdoms() function to retrieve
# kingdom information.
getKingdoms(db = "refseq")

# In this example, we will choose the bacteria kingdom.
# Now, the getGroups() function allows users to obtain available
# subgroups of the bacteria kingdom.
getGroups(db = "refseq", kingdom = "bacteria")

# Now we choose the group Gammaproteobacteria and specify
# the group argument in the meta.retrieval() function
meta.retrieval(kingdom = "bacteria",
   roup = "Gammaproteobacteria",
   db = "refseq",
```

```
    type = "genome")

## End(Not run)
```

`meta.retrieval.all` *Perform Meta-Genome Retrieval of all organisms in all kingdoms of life*

Description

Download genomes, proteomes, cds, gff, rna, or assembly stats files of individual species of all kingdoms of life.

Usage

```
meta.retrieval.all(db = "refseq", type = "genome", reference = FALSE)
```

Arguments

db	a character string specifying the database from which the genome shall be retrieved: • db = "refseq" • db = "genbank" • db = "emsembl" • db = "ensemblgenomes"
type	type of sequences that shall be retrieved. Options are: • type = "genome" : for genome assembly retrieval; see also getGenome), • type = "proteome" : (for proteome retrieval; see also getProteome), • type = "cds" : (for coding sequence retrieval; see also getCDS), • type = "gff" : (for annotation file retrieval in gff format; see also getGFF), • type = "gtf" : (for annotation file retrieval in gtf format (only for ensembl and ensemblgenomes); see also getGTF), • type = "rna" : (for RNA file retrieval in fasta format; see also getRNA), • type = "rm" : (for Repeat Masker output file retrieval; see also getRepeatMasker), • type = "assemblystats" (for genome assembly quality stats file retrieval; see also getAssemblyStats).
reference	a logical value indicating whether or not a genome shall be downloaded if it isn't marked in the database as either a reference genome or a representative genome. Options are: • reference = FALSE (Default): all organisms (reference, representative, and non-representative genomes) are downloaded. • reference = TRUE: organisms that are downloaded must be either a reference or representative genome. Thus, most genomes which are usually non-reference genomes will not be downloaded.

Details

This function aims to perform bulk retrieval of all genomes of species for all kingdoms of life.

Value

a character vector storing the file paths of the retrieved files.

Author(s)

Hajk-Georg Drost

See Also

meta.retrieval

Examples

```
## Not run:
# download all genomes from refseq
meta.retrieval.all(db = "refseq", type = "genome")
# download all vertebrate genomes from genbank
meta.retrieval.all(db = "genbank", type = "genome")
# download all vertebrate genomes from ensemblgenomes
meta.retrieval.all(db = "genbank", type = "ensemblgenomes")

## End(Not run)
```

---

organismAttributes *Retrieve Ensembl Biomart attributes for a query organism*

Description

In addition to the organismBM function, this function returns all available attributes that can be accessed through different marts and datasets for a given query organism.

Usage

```
organismAttributes(organism, update = FALSE, topic = NULL)
```

Arguments

organism	a character string specifying the scientific name of a query organism.
update	a logical value specifying whether or not the local listMart.txt, listDatasets.txt, and listAttributes_organism.txt files shall be updated by remote access to BioMart.
topic	a character string specifying a topic (category) of attributes, e.g. topic = "id".

Details

For a given query organism, this function retrieves all available attributes that can be accessed through different marts and datasets.

Sometimes the same attribute names correspond to different datasets and marts causing problems when using `getMarts`. The approach introduced by this function provides (again) a organism centric way of accessing organism specific attributes.

The `topic` argument allows the user to search for specific attribute topics/categories for faster filtering.

Value

a data.frame storing corresponding attribute names, description, datasets, and marts.

Note

When you run this function for the first time, the data retrieval procedure will take some time, due to the remote access to BioMart. The corresponding result is then saved in a *.txt file within the `tempdir` directory named "_biomart/listMarts.txt","_biomart/listDatasets.txt", and "_biomart/listAttributes_organism.txt", allowing subsequent queries to perform much faster.

Author(s)

Hajk-Georg Drost

References

http://biomart.org/

Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Steffen Durinck, Paul T. Spellman, Ewan Birney and Wolfgang Huber, Nature Protocols 4, 1184-1191 (2009).

BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. Steffen Durinck, Yves Moreau, Arek Kasprzyk, Sean Davis, Bart De Moor, Alvis Brazma and Wolfgang Huber, Bioinformatics 21, 3439-3440 (2005).

See Also

`organismFilters`, `organismBM`, `biomart`, `listAttributes`

Examples

```
## Not run:
# search for attribute topic id
head(organismAttributes("Homo sapiens", topic = "id"), 20)

## End(Not run)
```

organismBM *Retrieve Ensembl Biomart marts and datasets for a query organism*

Description

This function returns either all available biomart connections for all available organisms for which biomart access is possible, or (when specified) returns all organism specific biomart connections.

Usage

```
organismBM(organism = NULL, update = FALSE)
```

Arguments

organism	a character string specifying the scientific name of a query organism. Default is organism = NULL. In this case all available biomart connections are returned.
update	a logical value specifying whether or not the local listMart.txt and listDatasets.txt files shall be updated by remote access to BioMart.

Details

This function collects all available biomart connections and returns a table storing the organism for which biomart connections are available as well as the corresponding mart and database.

Note

When you run this function for the first time, the data retrieval procedure will take some time, due to the remote access to BioMart. The corresponding result is then saved in a *.txt file named "_biomart/listDatasets.txt" in the tempdir directory, allowing subsequent queries to perform much faster.

Author(s)

Hajk-Georg Drost

References

http://biomart.org/

Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Steffen Durinck, Paul T. Spellman, Ewan Birney and Wolfgang Huber, Nature Protocols 4, 1184-1191 (2009).

BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. Steffen Durinck, Yves Moreau, Arek Kasprzyk, Sean Davis, Bart De Moor, Alvis Brazma and Wolfgang Huber, Bioinformatics 21, 3439-3440 (2005).

See Also

getMarts, getDatasets, biomart, organismFilters, organismAttributes

Examples

```
## Not run:
# returning all available biomart connections
head(organismBM(), 20)
# retrieving all available datasets and biomart connections for
# a specific query organism (scientific name)
organismBM(organism = "Homo sapiens")
# you can also update the downloaded version using
# the "update = TRUE" argument
head(organismBM(update = TRUE), 20)

## End(Not run)
```

`organismFilters` *Retrieve Ensembl Biomart filters for a qyery organism*

Description

In addition to the `organismBM` and `organismAttributes` functions, this function returns all available filters that can be accessed through different marts and datasets for a given query organism.

Usage

```
organismFilters(organism, update = FALSE, topic = NULL)
```

Arguments

`organism`	a character string specifying the scientific name of a query organism.
`update`	a logical value specifying whether or not the local listMart.txt, listDatasets.txt, and listFilters_organism.txt files shall be updated by remote access to BioMart.
`topic`	a character string specifying a topic (category) of filters, e.g. `topic = "id"`.

Details

For a given query organism, this function retrieves all available filters that can be accessed through different marts and datasets.

Sometimes the same filter names correspond to different datasets and marts causing problems when using `getMarts`. The approach introduced by this function provides (again) a organism centric way of accessing organism specific filters.

The `topic` argument allows the user to search for specific filters topics/categories for faster selection.

Value

a data.frame storing corresponding filter names, description, datasets, and marts.

Note

When you run this function for the first time, the data retrieval procedure will take some time, due to the remote access to BioMart. The corresponding result is then saved in a *.txt file within the tempdir directory named "_biomart/listMarts.txt","_biomart/listDatasets.txt", and "_biomart/listFilters_organism.txt", allowing subsequent queries to perform much faster.

Author(s)

Hajk-Georg Drost

References

http://biomart.org/

Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Steffen Durinck, Paul T. Spellman, Ewan Birney and Wolfgang Huber, Nature Protocols 4, 1184-1191 (2009).

BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. Steffen Durinck, Yves Moreau, Arek Kasprzyk, Sean Davis, Bart De Moor, Alvis Brazma and Wolfgang Huber, Bioinformatics 21, 3439-3440 (2005).

See Also

organismBM, organismAttributes, getAttributes, getDatasets, getMarts

Examples

```
## Not run:
# search for filter topic "id"
head(organismFilters("Homo sapiens", topic = "id"), 20)

## End(Not run)
```

---

read_assemblystats *Import Genome Assembly Stats File*

Description

This function reads an organism specific Genome Assembly Stats file that was retrieved with getAssemblyStats.

Usage

```
read_assemblystats(file, type = "raw")
```

Arguments

file	a character string specifying the path to the file storing the Genome Assembly Stats file.
type	either type = "raw" to import the entire genome assembly stats file or type = "stats" to import overall statistics including all chromosomes, mitochondria and plastids.

Details

This function takes a string specifying the path to the Genome Assembly Stats file of interest (e.g. the path returned by getAssemblyStats) and imports it.

Author(s)

Hajk-Georg Drost

See Also

getAssemblyStats, read_genome, read_proteome, read_cds, read_gff

---

read_cds *Import CDS as Biostrings or data.table object*

Description

This function reads an organism specific CDS stored in a defined file format.

Usage

```
read_cds(
  file,
  format = "fasta",
  obj.type = "Biostrings",
  delete_corrupt = FALSE,
  ...
)
```

Arguments

file	a character string specifying the path to the file storing the CDS.
format	a character string specifying the file format used to store the genome, e.g. format = "fasta" (default) or format = "gbk".
obj.type	a character string specifying the object stype in which the genomic sequence shall be represented. Either as obj.type = "Biostrings" (default) or as obj.type = "data.table".

delete_corrupt	a logical value specifying whether potential CDS sequences that cannot be divided by 3 shall be be excluded from the the dataset. Default is delete_corrupt = FALSE.
...	additional arguments that are used by read.fasta.

Details

The read.cds function takes a string specifying the path to the cds file of interest as first argument.

It is possible to read in different proteome file standards such as *fasta* or *genebank*.

CDS stored in fasta files can be downloaded from http://www.ensembl.org/info/data/ftp/index.html.

Value

A data.table storing the gene id in the first column and the corresponding sequence as string in the second column.

Author(s)

Hajk-Georg Drost

See Also

getCDS, read_genome, read_proteome, read_gff, read_rna

---

read_genome *Import Genome Assembly as Biostrings or data.table object*

Description

This function reads an organism specific genome stored in a defined file format.

Usage

```
read_genome(file, format = "fasta", obj.type = "Biostrings", ...)
```

Arguments

file	a character string specifying the path to the file storing the genome.
format	a character string specifying the file format used to store the genome, e.g. format = "fasta" (default) or format = "gbk".
obj.type	a character string specifying the object stype in which the genomic sequence shall be represented. Either as obj.type = "Biostrings" (default) or as obj.type = "data.table".
...	additional arguments that are used by the read.fasta function.

Details

This function takes a string specifying the path to the genome file of interest as first argument (e.g. the path returned by `getGenome`).

Value

Either a `Biostrings` or `data.table` object.

Author(s)

Hajk-Georg Drost

See Also

`getGenome`, `read_proteome`, `read_cds`, `read_gff`, `read_rna`

`read_gff` *Import GFF File*

Description

This function reads an organism specific CDS stored in a defined file format.

Usage

```
read_gff(file)
```

Arguments

`file`	a character string specifying the path to the file storing the CDS.

Details

This function takes a string specifying the path to the GFF file of interest (e.g. the path returned by `getGFF`).

Value

Either a `Biostrings` or `data.table` object.

Author(s)

Hajk-Georg Drost

See Also

`getGenome`, `read_genome`, `read_proteome`, `read_cds`, `read_rna`

`read_proteome` *Import Proteome as Biostrings or data.table object*

Description

This function reads an organism specific proteome stored in a defined file format.

Usage

```
read_proteome(file, format = "fasta", obj.type = "Biostrings", ...)
```

Arguments

`file`	a character string specifying the path to the file storing the proteome.
`format`	a character string specifying the file format used to store the genome, e.g. `format = "fasta"` (default) or `format = "gbk"`.
`obj.type`	a character string specifying the object stype in which the genomic sequence shall be represented. Either as `obj.type = "Biostrings"` (default) or as `obj.type = "data.table"`.
`...`	additional arguments that are used by `read.fasta`.

Details

This function takes a string specifying the path to the proteome file of interest as first argument.

It is possible to read in different proteome file standards such as *fasta* or *genebank*.

Value

Either a `Biostrings` or `data.table` object.

Author(s)

Hajk-Georg Drost

See Also

`getProteome`, `read_genome`, `read_gff`, `read_cds`, `read_rna`

read_rm *Import Repeat Masker output file*

Description

This function reads an organism specific Repeat Masker output file.

Usage

```
read_rm(file)
```

Arguments

file	a character string specifying the path to the file storing the Repeat Masker output (e.g. retrieved with getRepeatMasker).

Details

This function takes a string specifying the path to the Repeat Masker output file of interest as first argument.

Author(s)

Hajk-Georg Drost

See Also

getRepeatMasker, read_genome, read_proteome, read_gff, read_rna

read_rna *Import RNA as Biostrings or data.table object*

Description

This function reads an organism specific RNA stored in a defined file format.

Usage

```
read_rna(file, format = "fasta", obj.type = "Biostrings", ...)
```

Arguments

file	a character string specifying the path to the file storing the RNA.
format	a character string specifying the file format used to store the genome, e.g. format = "fasta" (default) or format = "gbk".
obj.type	a character string specifying the object stype in which the genomic sequence shall be represented. Either as obj.type = "Biostrings" (default) or as obj.type = "data.table".
...	additional arguments that are used by read.fasta.

Details

This function takes a string specifying the path to the RNA file of interest as first argument. It is possible to read in different proteome file standards such as *fasta* or *genebank*.

Value

A data.table storing the gene id in the first column and the corresponding sequence as string in the second column.

Author(s)

Hajk-Georg Drost

See Also

getRNA, read_genome, read_proteome, read_gff

---

refseqOrganisms *Retrieve All Organism Names Stored on refseq*

Description

This function extracts all organism names (scientific names) for which genomes, proteomes, and CDS files are stored on the NCBI refseq server.

Usage

```
refseqOrganisms()
```

Author(s)

Hajk-Georg Drost

summary_cds *Retrieve summary statistics for a coding sequence (CDS) file*

Description

A summary statistics of specific CDS features is returned.

Usage

```
summary_cds(file, organism)
```

Arguments

file	file path to a CDS file in fasta format.
organism	character string specifying the organism at hand.

Details

The summary statistics include:

- total_seqs:
- nnn_abs: The total number of NNN's (over all chromosomes/scaffolds/contigs) in all coding sequences combined
- nnn_perc: The percentage (relative frequency) of NNN's (over all chromosomes/scaffolds/contigs) compared to the total number of nucleotides of all coding sequences

Author(s)

Hajk-Georg Drost

See Also

getCollection, getCDS, read_cds, summary_genome

summary_genome *Retrieve summary statistics for a genome assembly file*

Description

A summary statistics of specific genome features is generated. These statistics are useful to assess the genome quality of retrieved genome assemblies when performing comparative genomics tasks. This way, users can assess whether or not patterns found based on genome comparisons aren't just a technical artifact of differences in genome assembly quality.

Usage

```
summary_genome(file, organism)
```

Arguments

file	file path to a genome assembly file in fasta format.
organism	character string specifying the organism at hand.

Details

The summary statistics include:

- genome_size_mbp: Genome size in mega base pairs
- n50_mbp: The N50 contig size of the genome assembly in mega base pairs
- n_seqs: The number of chromosomes/scaffolds/contigs of the genome assembly file
- n_nnn: The absolute number of NNNs (over all chromosomes or scaffolds or contigs) in the genome assembly file
- rel_nnn: The percentage (relative frequency) of NNNs (over all chromosomes or scaffolds or contigs) compared to the total number of nucleotides in the genome assembly file
- genome_entropy: The Shannon Entropy of the genome assembly file (median entropy over all individual chromosome entropies)
- n_gc: The total number of GCs (over all chromosomes or scaffolds or contigs) in the genome assembly file
- rel_gc: The (relative frequency) of GCs (over all chromosomes or scaffolds or contigs) compared to the total number of nucleotides in the genome assembly file

Author(s)

Hajk-Georg Drost

See Also

summary_cds, getCollection, getGenome, read_genome

Examples

```
## Not run:
# retrieve genome from NCBI RefSeq
Sc <- biomartr::getGenome(db = "refseq", organism = "Saccharomyces cerevisiae")
# compute genome assembly summary statistics
Sc_genome_summary <- summary_genome(file = Sc, organism = "Saccharomyces cerevisiae")
# look at results
Sc_genome_summary

## End(Not run)
```

Index

∗ **package**
biomartr-package, 3

biomart, 4, *32, 33*, *62, 63*
biomartr *(*biomartr-package*)*, 3
biomartr-package, 3

check_annotation_biomartr, 5
clean.retrieval, 6, *57*, *59*

data.frame, *58*
download.database, 7, *8, 9*, *51, 52*
download.database.all, *8*, 8, *52*

getAssemblyStats, 9, *13*, *25*, *30*, *35*, *41*, *47*, *58*, *60*, *65, 66*
getAttributes, 11, *19*, *23*, *37*, *65*
getBM, *4, 5*, *33*
getCDS, *10*, 12, *15, 16*, *18*, *25*, *28*, *30*, *33*, *35*, *37*, *41*, *43*, *46, 47*, *49*, *58*, *60*, *67*, *72*
getCDSSet, 13, *16*, *18*, *28*, *32*, *43*
getCollection, 15, *18*, *59*, *72, 73*
getCollectionSet, 17
getDatasets, *11*, 19, *23*, *33*, *37*, *63*, *65*
getENSEMBL.gtf, 20
getENSEMBL.Seq, 21
getENSEMBLGENOMESInfo, 22
getENSEMBLInfo, 22
getFilters, *11*, *19*, 23, *33*, *37*
getGenome, *10*, *13*, *16*, *18*, 24, *30*, *33*, *35*, *37*, *41*, *46, 47*, *58*, *60*, *68*, *73*
getGENOMEREPORT, 26
getGenomeSet, *15, 16*, *18*, *25*, 26, *32*, *43*, *49*
getGFF, *10*, *13*, *15, 16*, *18*, *25*, *28*, 28, *32*, *35*, *38*, *41*, *43*, *46, 47*, *49*, *58*, *60*, *68*
getGFFSet, *15*, *28*, 30, *43*, *49*
getGO, 32
getGroups, 33, *37*, *58*
getGTF, *28*, 34, *47*, *58*, *60*
getKingdomAssemblySummary, 36, *40*, *50*
getKingdoms, *33*, 36, *50*, *54*, *58*
getMarts, *4*, *11*, *19*, *23*, *33*, 37, *62–65*
getMetaGenomeAnnotations, 38, *39*
getMetaGenomes, *38, 39*, 39, *56*
getMetaGenomeSummary, *36*, 40, *50*, *56*
getProteome, *10*, *13*, *16*, *18*, *25*, *30*, *33*, *35*, *37*, 40, *46, 47*, *58*, *60*, *69*
getProteomeSet, *15, 16*, *18*, *28*, *32*, 42, *49*
getReleases, 44
getRepeatMasker, *13*, *25*, *30*, *35*, *41*, 45, *47*, *58*, *60*, *70*
getRNA, *10*, *13*, *15, 16*, *18*, *25*, *28*, *30*, *32*, *35*, *41*, *43*, *46*, 46, *49*, *58*, *60*, *71*
getRNASet, *15*, *28*, *32*, *43*, 48, *49*
getSummaryFile, *36*, *40*, 49

is.genome.available, 50, *54, 55*

listAttributes, *5*, *62*
listDatabases, *7, 8*, 51
listGenomes, *51*, 52, *54, 55*
listGroups, 54, *55*
listKingdoms, *54*, 55
listMetaGenomes, *38, 39*, 56
listNCBIDatabases, *9*
listNCBIDatabases *(*listDatabases*)*, 51

meta.retrieval, *6, 7*, *10*, *13*, *15, 16*, *18*, *25*, *28*, *30*, *32, 33*, *35*, *37*, *41*, *43*, *46, 47*, *49*, 57, *61*
meta.retrieval.all, *25*, *59*, 60

organismAttributes, *11*, *19*, *23*, *37*, 61, *63–65*
organismBM, *5*, *11*, *19*, *23*, *33*, *37*, *61, 62*, 63, *64, 65*
organismFilters, *5*, *11*, *19*, *23*, *33*, *37*, *62, 63*, 64

read.fasta, *67*, *69*, *71*
read_assemblystats, *10*, 65

read_cds, *13*, *15*, *32*, *47*, *66*, 66, *68*, *69*, *72*
read_genome, *16*, *18*, *25*, *28*, *66*, *67*, 67, *68–71*, *73*
read_gff, *66–68*, 68, *69–71*
read_proteome, *41*, *43*, *66–68*, 69, *70*, *71*
read_rm, *46*, 70
read_rna, *49*, *67–70*, 70
refseqOrganisms, 71

summary_cds, 72, *73*
summary_genome, *24*, *25*, *72*, 72

tempdir, *6*, *16*, *18*, *29*, *31*, *34*, *62*, *63*, *65*