Package 'biomartr'

June 27, 2018

Title Genomic Data Retrieval

Version 0.8.0

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', 'ENSEMBLGENOMES', 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. as well as 'ENSEMBL' and 'ENSEMBLGENOMES' 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 (>= 0.2.2), downloader (>= 0.3), RCurl (>= 1.95-4.5), XML (>= 3.98-1.1), httr (>= 0.6.1), stringr (>= 0.6.2), purrr

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

License GPL-2

LazyData true

URL https://github.com/ropensci/biomartr

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

RoxygenNote 6.0.1

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 <hgd23@cam.ac.uk>

Repository CRAN

Date/Publication 2018-06-27 17:14:11 UTC

R topics documented:

`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 <hgd23@cam.ac.uk>

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, e.g. mart = "ensembl".
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)
```

`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 = TRUE,
  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" • db = "ensemblgenomes"
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 funcion 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 = TRUE,
  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" • db = "ensemblgenomes"
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.
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, 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)
```

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

Arguments

`db`	a character string specifying the database from which the collection shall be retrieved: • `db = "refseq"` • `db = "genbank"` • `db = "ensembl"` • `db = "ensemblgenomes"`
`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.

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

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

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("_ncbi_downloads","collection"))

## 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)
```

---

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

Examples

```
info.file <- getENSEMBLInfo()
info.file
```

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

```
# 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)
```

---

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.

Usage

```
getGenome(db = "refseq", organism, reference = TRUE,
  path = file.path("_ncbi_downloads", "genomes"))
```

Arguments

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

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

getProteome, getCDS, getGFF, getRNA, meta.retrieval, 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)
```

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 = TRUE,
  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"
- db = "ensemblgenomes"

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 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, 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)
```

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*

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, path = file.path("ensembl", "annotation"))
```

Arguments

`db`	a character string specifying the database from which the genome shall be retrieved: • `db = "ensembl"` • `db = "ensemblgenomes"`
`organism`	a character string specifying the scientific name of the organism of interest, e.g. `organism = "Homo sapiens"`.
`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")`.

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 '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`, `meta.retrieval`, `getGFF`

Examples

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


## 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

```
# get a table of all available databases from Ensembl Biomart
 getMarts()
```

---

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,
  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 = "ensemblgenomes" • 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.
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, 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)
```

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

Description

Retrieve available database releases or versions of ENSEMBL and ENSEMBLGENOMES.

Usage

```
getReleases(db = "refseq")
```

Arguments

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

Author(s)

Hajk-Georg Drost

Examples

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

## 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 = TRUE,
  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 = TRUE,
  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" or db = "ensemblgenomes".
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.
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, 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)
```

---

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 = "ensemblgenomes" • 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

```
# 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")
```

---

listGenomes *List All Available Genomes*

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, e.g. db = "refseq", db = "genbank", db = "ensembl", db = "ensemblgenomes".
type	a character string specifying a potential filter of available genomes. 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")

## End(Not run)
```

---

listGroups *List number of available genomes in each 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, e.g. `db = "refseq"` and `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.

Usage

```
meta.retrieval(db = "refseq", kingdom, group = NULL, type = "genome",
  restart_at_last = TRUE, reference = TRUE, 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" • db = "ensemblgenomes"
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"
- For ENSEMBLGENOMES
  - kingdom = "EnsemblBacteria"
  - kingdom = "EnsemblFungi"
  - kingdom = "EnsemblMetazoa"
  - kingdom = "EnsemblPlants"
  - kingdom = "EnsemblProtists"

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.

`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`

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")

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

```
                db = "ensemblgenomes",
                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")
```

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`).

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

Index