# Package 'biomartr'

August 29, 2016

**Title** Functional Annotation and Biological Data Retrieval with R

**Version** 0.1.0

**Author** Hajk-Georg Drost

**Maintainer** Hajk-Georg Drost <hgd23@cam.ac.uk>

**Description** Perform biological data retrieval and functional annotation with R. This tool provides API functions to the BioMart database and genome retrieval functions for bulk retrieval of genomes.

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Depends** R (>= 3.1.1)

**Imports** biomaRt, Biostrings, 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)

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

**License** GPL-3

**LazyData** true

**URL** https://github.com/HajkD/biomartr

**BugReports** https://github.com/HajkD/biomartr/issues

**RoxygenNote** 5.0.1

**Repository** CRAN

**Date/Publication** 2016-08-07 18:25:18

## R topics documented:

---

`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:
# the initial biomaRt workflow would work as follows:

# 1) select a mart and data set
mart <- useDataset("athaliana_eg_gene", mart = useMart("plants_mart_25"))

# 2) run a biomart query using the getBM() function
# and specify the attributes and filter arguments
geneSet <- c("AT1G06090", "AT1G06100",
             "AT1G06110", "AT1G06120",
             "AT1G06130", "AT1G06200")

resultTable <- getBM(attributes = c("start_position","end_position","description"),
                     filters = "tair_locus", values = geneSet, mart = mart)

# for faster query access and easier query logic the
# biomart() function combines this workflow

# using mart: 'plants_mart_25', 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_25",
        dataset    = "athaliana_eg_gene",
        attributes = c("start_position","end_position","description"),
        filters    = "tair_locus")
```

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

---

download_database *Download a NCBI Database to Your Local Hard Drive*

---

## Description

This function allows you to download a database selected by listDatabases to your local hard drive.

## Usage

```
download_database(name, db_format = "blastdb", path = "DB")
```

## Arguments

| | |
|---|---|
| name | a character string specifying the database that shall be downloaded (selected from listDatabases). |
| db_format | a character string specifying database format, e.g. db_format = "blastdb" or db_format = "fasta". |
| path | a character string specifying the location (a folder) in which the corresponding database shall be stored. Default is path = "DB". 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 DB (default) is created and the correspondning database then stored in this folder.

## Author(s)

Hajk-Georg Drost

## Examples

```
## Not run:

  # search for available NCBI nr databases
  listDatabases(db_name = "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(name = "nr.27.tar.gz", db_format = "blastdb")


## 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

This function retrieves a fasta-file storing the CDS files of the genome of an organism of interest and stores this file in the folder '_ncbi_downloads/CDS'.

## Usage

```
getCDS(db = "refseq", kingdom, organism, path = file.path("_ncbi_downloads",
  "CDS"), delete_corrupt = FALSE)
```

## Arguments

| | |
|---|---|
| db | a character string specifying the database from which the CDS file shall be retrieved: refseq. Right now only the ref seq database is included. Later version of **biomartr** will also allow sequence retrieval from additional databases. |
| kingdom | a character string specifying the kingdom of the organisms of interest, e.g. "archaea","bacteria", "fungi", "invertebrate", "plant", "protozoa", "vertebrate_mammalian", or "vertebrate_other". |
| organism | a character string specifying the scientific name of the organism of interest, e.g. 'Arabidopsis thaliana'. |
| 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"). |
| 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. |

## Details

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

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

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

## Value

A data.table storing the geneids in the first column and the DNA dequence in the second column.

## Author(s)

Hajk-Georg Drost

## References

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

http://www.ncbi.nlm.nih.gov/refseq/about/

## See Also

getGenome, getProteome, 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'
getCDS( db       = "refseq",
        kingdom  = "plant",
        organism = "Arabidopsis thaliana",
        path     = file.path("_ncbi_downloads","CDS"))


file_path <- file.path("_ncbi_downloads","CDS","Arabidopsis_thaliana_rna.fna.gz")
Ath_CDS <- read_cds(file_path, format = "fasta")


## 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)
```

---

# 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

This function retrieves a fasta-file storing the genome of an organism of interest and stores the genome file in the folder '_ncbi_downloads/genomes'.

## Usage

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

## Arguments

| | |
|---|---|
| `db` | a character string specifying the database from which the genome shall be retrieved: `refseq` or `genbank`. Right now only the ref seq database is included. Later version of **biomartr** will also allow sequence retrieval from additional databases. |
| `kingdom` | a character string specifying the kingdom of the organisms of interest, e.g. "archaea","bacteria", "fungi", "invertebrate", "plant", "protozoa", "vertebrate_mammalian", or "vertebrate_other". |
| `organism` | a character string specifying the scientific name of the organism of interest, e.g. 'Arabidopsis thaliana'. |
| `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

A data.table storing the geneids in the first column and the DNA dequence in the second column.

## Author(s)

Hajk-Georg Drost

## References

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

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

http://www.ncbi.nlm.nih.gov/refseq/about/

## See Also

getProteome, getCDS, 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'
getGenome( db       = "refseq",
           kingdom  = "plant",
           organism = "Arabidopsis thaliana",
           path = file.path("_ncbi_downloads","genomes"))

file_path <- file.path("_ncbi_downloads","genomes","Arabidopsis_thaliana_genomic.fna.gz")
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'
getGenome( db       = "genbank",
           kingdom  = "plant",
           organism = "Arabidopsis thaliana",
           path = file.path("_ncbi_downloads","genomes"))

file_path <- file.path("_ncbi_downloads","genomes","Arabidopsis_thaliana_genomic.fna.gz")
```

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

## 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, database = "BioMart", email = NULL, ...)
```

## 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 BioMart. |
| filters | a character vector specifying the filter (query key) for the BioMart query, e.g. filter = "ensembl_gene_id". |
| database | a cahracter string specifying the database from which GO information shall be retrieved. Possible choices are: database = "BioMart" and database = "DAVID". |
| email | a character string specifying the email address for your DAVID account. This parameter is only used to query DAVID web services and by default is email = NULL. |
| ... | 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",
                             "AT1G06110", "AT1G06120",
                             "AT1G06130", "AT1G06200"),
                filters  = "tair_locus")

# look at the result
head(GO_tbl[ , c("tair_locus","go_accession","go_name_1006")])


## End(Not run)
```

---

`getKingdoms` *Retrieve available kingdoms of life stored in RefSeq*

---

## Description

A short list of kingdoms of life that are stored in the RefSeq database and that can be downloaded using e.g. `meta.retrieval`, `getGenome`, etc.

## Usage

```
getKingdoms()
```

## Author(s)

Hajk-Georg Drost

## See Also

`meta.retrieval`, `getGenome`

## Examples

```
getKingdoms()
```

getMarts *Retrieve All Available BioMart Databases*

## Description

This funcion queries the BioMart API and returns a table storing all available 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 BioMart
 getMarts()

## End(Not run)
```

getProteome *Proteome Retrieval*

## Description

This function retrieves a fasta-file storing the proteome of an organism of interest and stores the proteome file in the folder '_ncbi_downloads/proteomes'.

## Usage

```
getProteome(db = "refseq", kingdom, organism,
  path = file.path("_ncbi_downloads", "proteomes"))
```

## Arguments

| | |
|---|---|
| db | a character string specifying the database from which the proteome shall be retrieved: refseq or genbank. Right now only the ref seq database is included. Later version of **biomartr** will also allow sequence retrieval from additional databases. |
| kingdom | a character string specifying the kingdom of the organisms of interest, e.g. "archaea","bacteria", "fungi", "invertebrate", "plant", "protozoa", "vertebrate_mammalian", or "vertebrate_other". |
| organism | a character string specifying the scientific name of the organism of interest, e.g. 'Arabidopsis thaliana'. |
| 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. 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

A data.table storing the geneids in the first column and the protein dequence in the second column.

## Author(s)

Hajk-Georg Drost

## References

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

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

http://www.ncbi.nlm.nih.gov/refseq/about/

## See Also

getGenome, getCDS, 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'
getProteome( db       = "refseq",
```

```
              kingdom  = "plant",
              organism = "Arabidopsis thaliana",
              path     = file.path("_ncbi_downloads","proteomes") )


file_path <- file.path("_ncbi_downloads","proteomes","Arabidopsis_thaliana_protein.faa.gz")
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'
getProteome( db       = "genbank",
              kingdom  = "plant",
              organism = "Arabidopsis thaliana",
              path     = file.path("_ncbi_downloads","proteomes") )


file_path <- file.path("_ncbi_downloads","proteomes","Arabidopsis_thaliana_protein.faa.gz")
Ath_proteome <- read_proteome(file_path, format = "fasta")

## 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(organism, details = FALSE, database = "refseq")
```

## Arguments

| | |
|---|---|
| `organism` | a character string specifying the scientific name of the organism of interest, e.g. 'Arabidopsis thaliana'. |
| `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. |
| `database` | a character string specifying the database for which genome availability shall be checked, e.g. `database = "refseq"` or `database = "all"`. |

## 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

## References

ftp://ftp.ncbi.nlm.nih.gov/genomes/GENOME_REPORTS/overview.txt

## Examples

```
## Not run:

# checking whether the Arabidopsis thaliana genome is stored on NCBI
is.genome.available(organism = "Arabidopsis thaliana")

# and printing details
is.genome.available(organism = "Arabidopsis thaliana", details = TRUE)


## 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_name = "nr", db_format = "fasta", update = FALSE)
```

## Arguments

| | |
|---|---|
| `db_name` | a character string specifying the name of the database that shall be searched for. |
| `db_format` | a character string specifying the database format, e.g. `db_format = "fasta"`. |
| `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

### References

ftp://ftp.ncbi.nlm.nih.gov/blast/db/FASTA

### See Also

download_database

### Examples

```
# retrieve all versions of the NCBI 'nr' database that can be downloaded
# listDatabases(db_name = "nr", db_format = "fasta")

# analogous:
# listDatabases(db_name = "cdd", db_format = "fasta")
# listDatabases(db_name = "nt", db_format = "fasta")
# listDatabases(db_name = "gss", db_format = "fasta")
# listDatabases(db_name = "refseq_protein", db_format = "fasta")
```

---

## 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(kingdom = "all", details = FALSE, update = FALSE,
  database = "all")
```

### Arguments

| | |
|---|---|
| kingdom | a character string specifying a potential filter of available genomes: "all","Archaea", "Bacteria", "Eukaryota", "Viroids", "Viruses". |
| 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", "chrs","organelles","plasmids", and "bio_projects". |
| update | a logical value specifying whether or not the available organism table shall be updated from the NCBI server. Default is update = FALSE. |
| database | a character string specifying the database for which genome availability shall be checked, e.g. database = "refseq" or database = "all". |

## Details

Internally this function loads the the overview.txt file from NCBI: ftp://ftp.ncbi.nlm.nih.gov/genomes/GENOME_REPORTS/ 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.

## Value

A data.frame storing either the organism names (details = FALSE) or all information present on the NCBI database (details = TRUE).

## Note

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

## Author(s)

Hajk-Georg Drost

## References

ftp://ftp.ncbi.nlm.nih.gov/genomes/GENOME_REPORTS/overview.txt

## Examples

```
## Not run:

# the simplest way to retrieve all names of genomes stored within NCBI databases
head(listGenomes() , 5)

# show all details
head(listGenomes(details = TRUE) , 5)

# show all details only for Bacteria
head(listGenomes(kingdom = "Bacteria", details = TRUE) , 5)

# in case you are interested in the number of genomes available for each kingdom, run:

ncbi_genomes <- listGenomes(details = TRUE)
table(ncbi_genomes[ , "kingdoms"])

# analogous, if you are interested in the number of genomes available for each group, run:
ncbi_genomes <- listGenomes(details = TRUE)
table(ncbi_genomes[ , "group"])

# for subgroup
table(ncbi_genomes[ , "subgroup"])

# you can also limit your search to the refseq database
```

```
head(listGenomes(database = "refseq") , 20)

head(listGenomes(details=TRUE, database = "refseq") , 5)

head(listGenomes(kingdom = "Eukaryota", details = TRUE,database = "refseq") , 5)

# order by file size
library(dplyr)
head(arrange(ncbi_genomes, desc(file_size_MB)) , 5)


# you can also update the organism table using the 'update' argument
head(listGenomes(details = TRUE,update = TRUE) , 5)


## End(Not run)
```

---

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

---

## Description

Download genomes, proteomes, or CDS of all species within a kingdom of life.

## Usage

```
meta.retrieval(kingdom, db = "refseq", type = "genome", out.folder = NULL)
```

## Arguments

| | |
|---|---|
| `kingdom` | a character string specifying the kingdom of the organisms of interest, e.g. "archaea","bacteria", "fungi", "invertebrate", "plant", "protozoa", "vertebrate_mammalian", or "vertebrate_other". |
| `db` | a character string specifying the database from which the genome shall be retrieved: `refseq` or `genbank`. |
| `type` | type of sequences that shall be retrieved. Either `genome`, `proteome`, or `CDS`. |
| `out.folder` | 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 of species that belong to the same kingdom of life.

## Author(s)

Hajk-Georg Drost

## Examples

```
## Not run:
# get all available kingdoms
getKingdoms()

# download all vertebrate genomes from refseq
meta.retrieval(kingdom = "vertebrate_mammalian", db = "refseq", type = "genome")

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

## End(Not run)
```

---

`organismAttributes` *Retrieve Biomart Attributes for an 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:

# return available attributes for "Homo sapiens"
head(organismAttributes("Homo sapiens"), 20)

# search for attribute topic "id"
head(organismAttributes("Homo sapiens", topic = "id"), 20)

# search for attribute topic "homolog"
head(organismAttributes("Homo sapiens", topic = "homolog"), 20)


## End(Not run)
```

---

## organismBM *Retrieve Biomart Marts and Datasets for an 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 Biomart Filters for an 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:

# return available filters for "Homo sapiens"
head(organismFilters("Homo sapiens"), 20)

# search for filter topic "id"
head(organismFilters("Homo sapiens", topic = "id"), 20)


## End(Not run)
```

---

## read_cds *Read the CDS of a given Organism*

---

### Description

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

### Usage

```
read_cds(file, format, 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 CDS, e.g. "fasta", "gbk". |

| | |
|---|---|
| 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 the seqinr::read.fasta() function. |

## 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

## Examples

```
## Not run:
# reading a cds file stored in fasta format
Ath.cds <- read.cds(system.file('seqs/ortho_thal_cds.fasta', package = 'orthologr'),
                    format = "fasta")

## End(Not run)
```

---

# read_genome *Read the genome of a given Organism*

---

## Description

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

## Usage

```
read_genome(file, format, ...)
```

## 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. "fasta", "gbk". |
| ... | additional arguments that are used by the seqinr::read.fasta() function. |

### Details

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

It is possible to read in different genome file standards such as *fasta* or *genebank*. Genomes stored in fasta files can be downloaded from http://ensemblgenomes.org/info/genomes.

### 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

### Examples

```
## Not run:
# reading a genome stored in a fasta file
Ath.genome <- read.genome(system.file('seqs/ortho_thal_cds.fasta', package = 'orthologr'),
                          format = "fasta")

## End(Not run)
```

---

## `read_proteome` *Read the proteome of a given Organism*

---

### Description

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

### Usage

```
read_proteome(file, format, ...)
```

### 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 proteome, e.g. "fasta", "gbk". |
| `...` | additional arguments that are used by the seqinr::read.fasta() function. |

### Details

The `read.proteome` 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*.

Proteomes stored in fasta files can be downloaded from http://www.ebi.ac.uk/reference_proteomes.

### 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

### Examples

```
## Not run:
# reading a proteome stored in a fasta file
Ath.proteome <- read.proteome(system.file('seqs/ortho_thal_aa.fasta', package = 'orthologr'),
                              format = "fasta")

## End(Not run)
```

---

## `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

### References

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

# Index