biomaRt

April 19, 2010

`attributePages` *Gives a summary of the attribute pages*

Description

Attributes in BioMart databases are grouped together in attribute pages. The attributePages function gives a summary of the attribute categories and groups present in the BioMart. These page names can be used to display only a subset of the available attributes in the listAttributes function.

Usage

```
attributePages(mart)
```

Arguments

`mart`	object of class Mart, created with the useMart function.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart = useMart("ensembl", dataset="hsapiens_gene_ensembl")
attributeSummary(mart)
}
```

attributeSummary *Gives a summary of the attributes groups and categories*

Description

Attributes in BioMart databases are grouped together in attribute groups and attribute groups can in their turn form a higher collection in an attribute category. The attributeSummary function gives a summary of the attribute categories and groups present in the BioMart. These group and category names can be used to display only a subset of the available attributes in the listAttributes function. NOTE: pending on availability from the BioMart web service, groups are currently not available and this function will be replaced by the attributePages function to comply better with the BioMart naming scheme.

Usage

```
attributeSummary(mart)
```

Arguments

mart	object of class Mart, created with the useMart function.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart = useMart("ensembl", dataset="hsapiens_gene_ensembl")
attributeSummary(mart)
}
```

exportFASTA *Exports getSequence results to FASTA format*

Description

Exports getSequence results to FASTA format

Usage

```
exportFASTA(sequences, file)
```

Arguments

sequences	A data.frame that was the output of the getSequence function
file	File to which you want to write the data

Author(s)

Steffen Durinck

Examples

```
if(interactive()){
mart <- useMart("ensembl", dataset="hsapiens_gene_ensembl")

#seq<-getSequence(chromosome=c(2,2),start=c(100000,30000),end=c(100300,30500),mart=mart)
#exportFASTA(seq,file="test.fasta")

martDisconnect(mart = mart)
}
```

---

filterOptions *Displays the filter options*

Description

Displays a set of predetermed values for the specified filter (if available).

Usage

```
filterOptions(filter,mart)
```

Arguments

filter	A valid filter name.
mart	object of class Mar created using the useMart functiont

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart = useMart("ensembl", dataset="hsapiens_gene_ensembl")
filterOptions("chromosome_name", mart)
}
```

filterType *Displays the filter type*

Description

Displays the type of the filer given a filter name.

Usage

```
filterType(filter,mart)
```

Arguments

filter	A valid filter name. Valid filters are given by the listFilters function
mart	object of class Mart, created using the useMart function

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart = useMart("ensembl", dataset="hsapiens_gene_ensembl")
filterType("chromosome_name", mart)
}
```

getBMlist *Retrieves information from the BioMart database*

Description

This function is the main biomaRt query function. Given a set of filters and corresponding values, it retrieves the user specified attributes from the BioMart database one is connected to

Usage

```
getBMlist(attributes, filters = "", values = "", mart, list.names = NULL, na.val
```

Arguments

attributes	Attributes you want to retrieve. A possible list of attributes can be retrieved using the function listAttributes.
filters	Filters (one or more) that should be used in the query. A possible list of filters can be retrieved using the function listFilters.
values	Values of the filter, e.g. vector of affy IDs. If multiple filters are specified then the argument should be a list of vectors of which the position of each vector corresponds to the position of the filters in the filters argument.

`mart`	object of class Mart, created with the useMart function.
`list.names`	names for objects in list
`na.value`	value to give when result is empty
`verbose`	When using biomaRt in webservice mode and setting verbose to TRUE, the XML query to the webservice will be printed.
`giveWarning`	Gives a warning about best practices of biomaRt and recommends using getBM instead of getBMlist

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart <- useMart("ensembl")
datasets <- listDatasets(mart)

}
```

`getBM` *Retrieves information from the BioMart database*

Description

This function is the main biomaRt query function. Given a set of filters and corresponding values, it retrieves the user specified attributes from the BioMart database one is connected to

Usage

```
getBM(attributes, filters = "", values = "", mart, curl = NULL, checkFilters = T
```

Arguments

`attributes`	Attributes you want to retrieve. A possible list of attributes can be retrieved using the function listAttributes.
`filters`	Filters (one or more) that should be used in the query. A possible list of filters can be retrieved using the function listFilters.
`values`	Values of the filter, e.g. vector of affy IDs. If multiple filters are specified then the argument should be a list of vectors of which the position of each vector corresponds to the position of the filters in the filters argument.
`mart`	object of class Mart, created with the useMart function.
`curl`	An optional 'CURLHandle' object, that can be used to speed up getBM when used in a loop.
`checkFilters`	Sometimes attributes where a value needs to be specified, for example upstream_flank with value 20 for obtaining upstream sequence flank regions of length 20bp, are treated as filters in BioMarts. To enable such a query to work, one must specify the attribute as a filter and set checkFilters = FALSE for the query to work.

verbose	When using biomaRt in webservice mode and setting verbose to TRUE, the XML query to the webservice will be printed.
uniqueRows	If the result of a query contains multiple identical rows, setting this argument to TRUE (default) will result in deleting the duplicated rows in the query result at the server side.
output	getBM always outputs a data.frame. Use the getBMlist function if a list is required, note that this is not the recommended way to query and getBM should be prefered.
list.names	getBM always outputs a data.frame. Use the getBMlist function if a list is required, note that this is not the recommended way to query and getBM should be prefered.
na.value	getBM always outputs a data.frame. Use the getBMlist function if a list is required, note that this is not the recommended way to query and getBM should be prefered.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart <- useMart("ensembl")
datasets <- listDatasets(mart)
mart<-useDataset("hsapiens_gene_ensembl",mart)
getBM(attributes=c("affy_hg_u95av2","hgnc_symbol","chromosome_name","band"),filters="affy

}
```

---

getGene *Retrieves gene annotation information given a vector of identifiers*

Description

This function retrieves gene annotations from Ensembl given a vector of identifiers. Annotation includes chromsome name, band, start position, end position, gene description and gene symbol. A wide variety of identifiers is available in Ensembl, these can be found with the listFilters function.

Usage

```
getGene( id, type, mart)
```

Arguments

id	vector of gene identifiers one wants to annotate
type	type of identifier, possible values can be obtained by the listFilters function. Examples are entrezgene, hgnc_symbol (for hugo gene symbol), ensembl_gene_id, unigene, agilentprobe, affy_hg_u133_plus_2, refseq_dna, etc.
mart	object of class Mart, containing connections to the BioMart databases. You can create such an object using the function useMart.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){

mart = useMart("ensembl", dataset="hsapiens_gene_ensembl")

#example using affy id

g = getGene( id = "1939_at", type = "affy_hg_u95av2", mart = mart)
show(g)

#example using Entrez Gene id

g = getGene( id = "100", type = "entrezgene", mart = mart)
show(g)
}
```

---

getGO *Retrieves GO information*

Description

This function is now defunct, use getBM instead. See vignette for an example to retrieve GO information from Ensembl using getBM..This function retrieves GO identifiers, GO descriptions and evidence codes from Ensembl given a vector of gene identifier. A wide variety of gene indentifiers can be used as inputs. The list of possible identifiers that can be used as input, can be found using the listFilters function.

Usage

```
getGO( id, type, mart)
```

Arguments

id	gene identifier
type	type of identifier, possible values can be obtained by the listFilters function. Examples are entrezgene, hgnc_symbol (for hugo gene sy mbol), ensembl_gene_id, unigene, agilentprobe, affy_hg_u133_plus_2, refseq_dna, etc.
mart	object of class Mart, containing connections to the BioMart databases. You can create such an object using the function useMart.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){

mart <- useMart("ensembl", dataset="hsapiens_gene_ensembl")

#example using affy id

go = getGO( id = "1939_at", type = "affy_hg_u95av2", mart = mart)
show(go)

#example using entrezgene id

go = getGO( id = 672, type = "entrezgene", mart = mart)
show(go)
}
```

---

getLDS *Retrieves information from two linked datasets*

Description

This function is the main biomaRt query function that links 2 datasets and retrieves information from these linked BioMart datasets. In Ensembl this translates to homology mapping.

Usage

```
getLDS(attributes, filters = "", values = "", mart, attributesL, filtersL = "",
```

Arguments

attributes	Attributes you want to retrieve of primary dataset. A possible list of attributes can be retrieved using the function listAttributes.
filters	Filters that should be used in the query. These filters will be applied to primary dataset. A possible list of filters can be retrieved using the function listFilters.
values	Values of the filter, e.g. list of affy IDs
mart	object of class Mart created with the useMart function.
attributesL	Attributes of linked dataset that needs to be retrieved
filtersL	Filters to be applied to the linked dataset
valuesL	Values for the linked dataset filters
martL	Mart object representing linked dataset
verbose	When using biomaRt in webservice mode and setting verbose to TRUE, the XML query to the webservice will be printed. Alternatively in MySQL mode the MySQL query will be printed.
uniqueRows	Logical to indicate if the BioMart web service should return unique rows only or not. Has the value of either TRUE or FALSE

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
human = useMart("ensembl", dataset = "hsapiens_gene_ensembl")
mouse = useMart("ensembl", dataset = "mmusculus_gene_ensembl")
getLDS(attributes = c("hgnc_symbol","chromosome_name", "start_position"), filters = "hgnc
}
```

---

getSequence *Retrieves sequences*

Description

This function retrieves sequences given the chomosome, start and end position or a list of identifiers. Using getSequence in web service mode (default) generates 5' to 3' sequences of the requested type on the correct strand. The type of sequence returned can be specified by the seqType argument which takes the following values: 'cdna';'peptide' for protein sequences;'3utr' for 3' UTR sequences,'5utr' for 5' UTR sequences; 'gene_exon' for exon sequences only; 'transcript_exon_intron' gives the full unspliced transcript, that is exons + introns;'gene_exon_intron' gives the exons + introns of a gene;'coding' gives the coding sequence only;'coding_transcript_flank' gives the flanking region of the transcript including the UTRs, this must be accompanied with a given value for the upstream or downstream attribute;'coding_gene_flank' gives the flanking region of the gene including the UTRs, this must be accompanied with a given value for the upstream or downstream attribute; 'transcript_flank' gives the flanking region of the transcript exculding the UTRs, this must be accompanied with a given value for the upstream or downstream attribute; 'gene_flank' gives the flanking region of the gene excluding the UTRs, this must be accompanied with a given value for the upstream or downstream attribute. In MySQL mode the getSequence function is more limited and the sequence that is returned is the 5' to 3'+ strand of the genomic sequence, given a chromosome, as start and an end position. So if the sequence of interest is the minus strand, one has to compute the reverse complement of the retrieved sequence, which can be done using functions provided in the matchprobes package. The biomaRt vignette contains more examples on how to use this function.

Usage

```
getSequence( chromosome, start, end, id, type, seqType, upstream, downstream, ma
```

Arguments

chromosome	Chromosome name
start	start position of sequence on chromosome
end	end position of sequence on chromosome
id	An identifier or vector of identifiers.
type	The type of identifier used. Supported types are hugo, ensembl, embl, entrezgene, refseq, ensemblTrans and unigene. Alternatively one can also use a filter to specify the type. Possible filters are given by the listFilters function
seqType	Type of sequence that you want to retrieve. Allowed seqTypes are: cdna, peptide, 3utr, 5utr, genomic
upstream	To add the upstream sequence of a specified number of basepairs to the output.

downstream	To add the downstream sequence of a specified number of basepairs to the output.
mart	object of class Mart created using the useMart function
verbose	If verbose = TRUE then the XML query that was send to the webservice will be displayed.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart <- useMart("ensembl",dataset="hsapiens_gene_ensembl")

seq = getSequence(id="BRCA1", type="hgnc_symbol", seqType="peptide", mart = mart)
show(seq)


seq = getSequence(id="1939_at", type="affy_hg_u95av2", seqType="gene_flank",upstream = 20
show(seq)

}
```

---

listAttributes *lists the attributes available in the selected dataset*

Description

Attributes are the outputs of a biomaRt query, they are the information we want to retrieve. For example if we want to retrieve all entrez gene identifiers of genes located on chromosome X, entrezgene will be the attribute we use in the query. The listAttributes function lists the available attributes in the selected dataset

Usage

```
listAttributes(mart, page,what = c("name","description"), group, category, showG
```

Arguments

mart	object of class Mart created using the useMart function
page	Show only the attributes that belong to the specified attribute page.
what	vector of types of information about the attributes that need to be displayed. Can have values like name, description, fullDescription, page
group	Availability of group argument is pending on availability from BioMart web service. Currently this argument can not be used
category	Category is now replaced by page to better comply with the BioMart suite http://www.biomart.org
showGroups	Availability of showGroups argument is pending on availability from BioMart web service. Currently this argument can not be used

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
ensembl = useMart("ensembl", dataset="hsapiens_gene_ensembl")
listAttributes(ensembl)
}
```

`listDatasets` *lists the datasets available in the selected BioMart database*

Description

Lists the datasets available in the selected BioMart database

Usage

```
listDatasets(mart)
```

Arguments

`mart`	object of class Mart created with the useMart function

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){

#marts <- listMarts()
#index<-grep("ensembl",marts)

#mart <- useMart(marts[index])

#listDatasets(mart = mart)

#martDisconnect(mart = mart)
}
```

listFilters *lists the filters available in the selected dataset*

Description

Filters are what we use as inputs for a biomaRt query. For example, if we want to retrieve all entrezgene identifiers on chromosome X, `chromosome` will be the filter, with corresponding value X.

Usage

```
listFilters(mart, what = c("name", "description"),
                    group = "DEFUNCT")
```

Arguments

`mart`	object of class `Mart` created using the `useMart` function
`what`	character vector indicating what information to display about the available filters. Valid values are `name`, `description`, `options`, `fullDescription`, `filters`, `type`, `operation`, `filters8`, `filters9`.
`group`	defunct. If you need advice how to replace that functionality, please contact the package maintainer for advice.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart = useMart("ensembl", dataset="hsapiens_gene_ensembl")
listFilters(mart)
}
```

listMarts *lists the avilable BioMart databases*

Description

This function returns a list of BioMart databases to which biomaRt can connect to. By default all public BioMart databases are displayed. To establish a connection use the useMart function.

Usage

```
listMarts(mart, host="www.biomart.org", path="/biomart/martservice", port=80, in
```

Arguments

mart	mart object created with the useMart function
host	host to connect to if different then www.biomart.org
path	path to martservice that should be pasted behind the host to get to web service URL
port	port to use in HTTP communication
includeHosts	boolean to indicate if function should return host of the BioMart databases
archive	Boolean to indicate if you want to access archived versions of BioMart database
user	MySQL access is no longer available through this package
password	MySQL access is no longer available through this package
mysql	MySQL access is no longer available through this package. All queries can be made through the web service mode. use mysql=FALSE

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
listMarts()
}
```

Mart-class *Class Mart*

Description

Represents a Mart class, containing connections to different BioMarts

Methods

show Print summary of the object

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

martDisconnect *Disconnects from BioMarts (defunct)*

Description

The function 'martDisconnect' is defunct. In BioMart webservice mode, disconnecting is not necessary. Background explanation: In previous versions of this package, a MySQL mode was also supported, which required disconnecting.

Arguments

mart	Mart object created with the useMart function

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart <- martConnect()

martDisconnect(mart = mart)
}
```

NP2009code *Display the analysis code from the 2009 Nature protocols paper*

Description

This function opens an editor displaying the analysis code of the Nature Protocols 2009 paper

Usage

```
NP2009code()
```

Details

The edit function uses getOption("editor") to select the editor. Use, for instance, options(editor="emac
to set another editor.

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen, Wolfgang Huber

See Also

edit

Examples

```
if(interactive()){
NP2009code()
}
```

`useDataset` *Select a dataset to use and updates Mart object*

Description

This function selects a dataset and updates the Mart object

Usage

```
useDataset(dataset,mart)
```

Arguments

`dataset`	Dataset you want to use. List of possible datasets can be retrieved using the function listDatasets
`mart`	Mart object created with the useMart function

Author(s)

Steffen Durinck, http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){
mart=useMart("ensembl")
mart=useDataset("hsapiens_gene_ensembl", mart = mart)
}
```

`useMart` *Connects to the selected BioMart database and dataset*

Description

A first step in using the biomaRt package is to select a BioMart database and dataset to use. The useMart function enables one to connect to a specified BioMart database and dataset within this database. To know which BioMart databases are available see the listMarts function. To know which datasets are available within a BioMart database, first select the BioMart database using useMart and then use the listDatasets function on the selected BioMart, see listDatasets function.

Usage

```
useMart(biomart, dataset, host="www.biomart.org", path="/biomart/martservice", p
```

Arguments

biomart	BioMart database name you want to connect to. Possible database names can be retrieved with the functio listMarts
dataset	Dataset you want to use. To see the different datasets available within a biomaRt you can e.g. do: mart = useMart('ensembl'), followed by listDatasets(mart).
host	Host to connect to if different then www.biomart.org
path	Path that should be pasted after to host to get access to the web service URL
port	port to connect to, will be pasted between host and path
archive	Boolean to indicate if you want to access archived versions od BioMart databases. This currently only works with mysql=TRUE and the ensembl BioMart database
local	MySQL access to Ensembl is no longer supported. biomaRt in web service mode supports all BioMart queries. Local BioMart databases can be accessed by specifying the local host which runs the local BioMart web service.
mysql	MySQL access to Ensembl is no longer supported. biomaRt in web service mode supports all BioMart queries. Local BioMart databases can be accessed by specifying the local host which runs the local BioMart web service.
user	MySQL access to Ensembl is no longer supported. biomaRt in web service mode supports all BioMart queries. Local BioMart databases can be accessed by specifying the local host which runs the local BioMart web service.
password	MySQL access to Ensembl is no longer supported. biomaRt in web service mode supports all BioMart queries. Local BioMart databases can be accessed by specifying the local host which runs the local BioMart web service.

Author(s)

Steffen Durinck,http://www.stat.berkeley.edu/~steffen

Examples

```
if(interactive()){

mart = useMart("ensembl")
mart=useMart(biomart="ensembl", dataset="hsapiens_gene_ensembl")
}
```

Index

∗Topic **methods**
attributePages, 1
attributeSummary, 2
exportFASTA, 2
filterOptions, 3
filterType, 4
getBM, 5
getBMlist, 4
getGene, 6
getGO, 7
getLDS, 8
getSequence, 9
listAttributes, 10
listDatasets, 11
listFilters, 12
listMarts, 12
Mart-class, 13
martDisconnect, 14
NP2009code, 14
useDataset, 15
useMart, 15

attributePages, 1
attributeSummary, 2

edit, *14*
exportFASTA, 2

filterOptions, 3
filterType, 4

getBM, 5
getBMlist, 4
getGene, 6
getGO, 7
getLDS, 8
getSequence, 9

listAttributes, 10
listDatasets, 11
listFilters, 12
listMarts, 12

Mart-class, 13
martDisconnect, 14

NP2009code, 14

show,Mart-method *(Mart-class)*, 13

useDataset, 15
useMart, *12*, 15