Package 'RInterMine'

October 7, 2014

Type Package

Title R Interface with InterMine-powered databases

Version 1.0

Date 2012-08-30

Author InterMine Team

Maintainer InterMine Team <info@intermine.org>

Description InterMine-powered databases such as FlyMine, modENCODE, RatMine, YeastMine, MetabolicMine and TargetMine are integrated databases of genomic, expression and protein data for various organisms. Integrating data makes it possible to run sophisticated data mining queries that span domains of biological knowledge. This R package provides interface with these databases through webservices. It makes most from the correspondence of the data frame object in R and the table object in databases while hiding the details of data exchange through XML or JSON.

License LGPL

Imports Biostrings, RCurl, XML, RJSONIO, sqldf, igraph

Suggests Gviz

R topics documented:

RInterMine-package . . . 2
deleteList . . . 2
doEnrichment . . . 3
getLists . . . 4
getModel . . . 5
getRegionFeature . . . 6
getRegionSequence . . . 7
getRelease . . . 7
getTemplateQuery . . . 8
getTemplates . . . 9
getVersion . . . 10
initInterMine . . . 10
listMines . . . 11
newList . . . 12

newQuery 12
renameList 13
runQuery 14

Index **16**

`RInterMine-package` *R Interface with InterMine-powered databases*

Description

InterMine-powered databases such as FlyMine, modENCODE, RatMine, YeastMine, MetabolicMine and TargetMine are integrated databases of genomic, expression and protein data for various organisms. Integrating data makes it possible to run sophisticated data mining queries that span domains of biological knowledge. This R package provides interface with these databases through webservices. It makes most from the correspondence of the data frame object in R and the table object in databases while hiding the details of data exchange through XML or JSON.

Details

Package:	RInterMine
Type:	Package
Version:	1.0
Date:	2012-09-04
License:	LGPL
Depends:	RCurl, XML, RJSONIO, sqldf, igraph

Author(s)

InterMine Team

Maintainer: InterMine Team <info@intermine.org>

References

http://intermine.org/wiki/WebService

`deleteList` *Delete a list stored on the mine*

Description

Delete a list stored on the mine.

Usage

```
deleteList(im, name, timeout=3)
```

Arguments

im	a list containing the base URL and API token.
name	a string, representing the name of the list to delete.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a list, represetning the response from the server for the delete operation.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta", "TOKEN") #replace TOKEN with your token

## Not run: status <- deleteList(im, "a") # "a" is the name of a list
```

---

doEnrichment *Do enrichment analysis for a list of genes*

Description

Do enrichment analysis for a list of genes.

Usage

```
doEnrichment(im, genelist, ontology, subcategory = "", maxp = 0.05,
correction = c("Holm-Bonferroni", "Benjamini and Hochberg", "Bonferroni", "None"),
timeout = 60)
```

Arguments

im	a list, containing the base URL and API token.
genelist	a character vector, represetning the genes for enrichment analysis
ontology	a string, representing the name of the ontology. It could be of one of the following values: "go_enrichment_for_gene" (for GO enrichment), "pathway_enrichment" (for pathway enrichment), "prot_dom_enrichment_for_gene" (for Protein domain enrichment), "publication_enrichment" (for Publication enrichment).
subcategory	a string, representing the sub-category of the ontology. It could be of the following values: "biological_process", "cellular_component", "molecular_function" (for GO enrichment) or "All", "KEGG pathways data set", "Reactome data set" (for pathway enrichment)
maxp	a numeric, representing the threshold p-value.
correction	a string, representing the correction method for multiple comparison. It could be of one of the following values: "Holm-Bonferroni", "Benjamini and Hochberg", "Bonferroni", "None".
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a matrix, representing the enrichment result.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta", "TOKEN") #replace TOKEN with your token

## Not run: doEnrichment(im, c("ABO", "ALB"), 'go_enrichment_for_gene', 'cellular_component',
      correction="NONE")
## End(Not run)
```

---

getLists *Get the information of the lists stored on the mine*

Description

Returns information on the lists the user has access to in the mine.

Usage

```
getLists(im, timeout = 3)
```

Arguments

im	a list, containing the base URL and API token.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a character vector, representing the names of the lists on the mine.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta", "TOKEN") #replace TOKEN with your token

gl <- getLists(im)
```

getModel *Get the model of InterMine*

Description

Returns a representation of the data model for the mine. This describes the kind of data held, and the properties that data can have. This information can be used to build queries against that data, and to interpret the information received.

Usage

```
getModel(im, timeout = 3)
```

Arguments

im	a list containing the base URL and API token.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Details

The details of the data model for the various mines are available at the websites obtained by running the following command: paste(listMines()$URL, "/tree.do", sep="")

Value

a multi-level list, representating the data model for the mine. The first-level is a list of the InterMine objects (e.g., Gene, Exon). Each second-level list, corresponding to an InterMine object, contains three data.frame objects: attributes, references and collections. Each attribute is a property of the InterMine object. Each reference or collection is itself an InterMine object, acting as a member object of the InterMine object.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta")

model <- getModel(im)
```

`getRegionFeature` *Obtain the features (exons, transcripts, genes) in a region of the genome in bed format*

Description

Obtain the features (exons, transcripts, genes) in a region of the genome in bed format

Usage

```
getRegionFeature(im, regions, featureType, organism = "H. sapiens", extension = 100,
      isInterbase = F, timeout = 60)
```

Arguments

`im`	a list containing the base URL and API token.
`regions`	a character vector, representing genomic regions, e.g., "X:99000000..99895088".
`featureType`	a character vector, represeting the type of features. It could be any of these values: "Gene", "Transcript", "Exon".
`organism`	a string, representing the name of the organism.
`extension`	an integer, representing how far, in base-pairs, to extend the regions on each side.
`isInterbase`	a boolean, representing whether to treat the region as interbase co-ordinates.
`timeout`	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a data.frame object, representing the information of the feature (exon, intron or gene) in bed format.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta") #replace TOKEN with your token

gf <- getRegionFeature(im, c("X:99000000..99895088"), c("Exon"))
```

getRegionSequence *Obtain the DNA sequence in regions of the genome*

Description

Obtain the DNA sequence in regions of the genome

Usage

```
getRegionSequence(im, regions, organism = "H. sapiens", extension = 100,
      isInterbase = F, timeout = 60)
```

Arguments

im	a list containing the base URL and API token.
regions	a character vector, representing genomic regions, e.g., "X:99000000..99895088".
organism	a string, representing the name of the organism.
extension	an integer, representing how far, in base-pairs, to extend the regions on each side.
isInterbase	a boolean, representing whether to treat the region as interbase co-ordinates.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a XstringSet object, representing the DNA sequences in the specified regions.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta")

gf <- getRegionSequence(im, c("X:99000000..99895088"))
```

getRelease *Get the current release information of InterMine*

Description

Returns a string describing the release of the mine.

Usage

```
getRelease(im, timeout = 3)
```

Arguments

im	a list containing the base URL and API token.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a string, describing the release of the mine.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta")

getRelease(im)
```

---

getTemplateQuery *Get the query contained in a template*

Description

Get a template query for a mine. A template contain a saved query with a view and constraint. The user can modify this query to obtain the desired result. The view is a vector containing the output columns of the query. The constraint is a matrix containing the following columns: path (the path of the constraint), op (the constraint operator, one of '=', '!=', 'LOOKUP', 'ONE OF', 'NONE OF', '>', '<', '>=', '<=', 'LIKE'), value (the constraint value), code (the name of the constraint), extraValue (optional, required for LOOKUP constraints).

Usage

```
getTemplateQuery(im, name, timeout=3)
```

Arguments

im	a list, containing the base URL and API token.
name	a string, representing the name of the pre-defined template.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a list, reprenting the query contained in the pre-defined template. The list should contain at least two elements, view and constrain.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta")

queryGeneIden <- getTemplateQuery(im, "Gene_Identifiers")
```

`getTemplates` *Get the information (name and title) of the templates pre-defined in InterMine*

Description

Get the information (name and title) of the templates pre-defined in InterMine. A template contain a query with fixed set of output columns, and at least one editable constraint, and possibly more.

Usage

```
getTemplates(im, format = "data.frame", timeout = 3)
```

Arguments

`im`	a list, containing the base URL and API token.
`format`	a string with values being either "data.frame" or "list", representing the output format of the template information.
`timeout`	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a data.frame or list object, representing the information (name and title) for the pre-define templates in the mine.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta")

template <- getTemplates(im)
```

getVersion *Get the version information of InterMine*

Description

Returns an integer representing the capabilities of the webservice.

Usage

```
getVersion(im, timeout = 3)
```

Arguments

im	a list containing the base URL and API token.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

an integer, representing the capabilities of the webservice.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta")
getVersion(im)
```

initInterMine *Initialize the list containing the base URL and API token.*

Description

Initialize the InterMine list with the base URL of the webservice of the database and the API token.

Some resources such as lists are normally privately associated with the individual user that created them and require authentication for access. To access these private resources, each request needs to be authenticated, using an API key token. You can get an API token from the web-app of the service you intend to access: visit the MyMine tab after logging-in and click on API Key.

Usage

```
initInterMine(mine = listMines()["MetabolicMine"], token="")
```

Arguments

mine	a string, representing the base URL of the webservice of the database.
token	a string, representing the API token in order to use private functions such as list and enrichment.

Value

A list, containing the base URL and API token.

Author(s)

InterMine Team

References

http://intermine.org/wiki/WebService#APIKeyTokens

Examples

```
im <- initInterMine(mine = listMines()["MetabolicMine"], "TOKEN") #replace TOKEN with your token
```

---

listMines *List the available InterMine-powered databases*

Description

InterMine-powered databases such as FlyMine, modENCODE, RatMine, YeastMine, MetabolicMine and TargetMine are integrated databases of genomic, expression and protein data for various organisms. The function listMines() lists the current available databases.

Usage

```
listMines()
```

Value

A data frame object containing two columns:

URL	the base URL of the webservice of the database
name	the name of the database

Author(s)

InterMine Team

References

http://intermine.org

Examples

```
listMines()
```

newList *Create a new list on the mine*

Description

Create a new list on the mine.

Usage

```
newList(im, name, gene,  organism="H.+sapiens",  description="", timeout=30)
```

Arguments

im	a list, containing the base URL and API token.
name	a string, representing the name of the list.
gene	a charater vector or a file name containing the genes in the list.
organism	a string, representing the organism which the genes are associated with.
description	a string, representing the description of the genes.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a list, representing the response from the server for the new operation.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta", "TOKEN") #replace TOKEN with your token

## Not run: gl <- newList(im, "a", c("ABO", "ALB"))
```

newQuery *Initialize a new query*

Description

A query needs to have at least view, constraints and constraintLogic. The view is a vector containing the columns of the query output. The constraint is a matrix containing the following columns: path (the path of the constraint), op (the constraint operator, one of '=', '!=', 'LOOKUP', 'ONE OF', 'NONE OF', '>', '<', '>=', '<=', 'LIKE'), value (the constraint value), code (the name of the constraint), extraValue (optional, required for LOOKUP constraints). The constraintLogic by default is "AND" operation, e.g., "A and B", where A and B are the codes in the constraints.

Usage

```
newQuery(name="", view=character(), sortOrder="", longDescription="",
      constraints=matrix(character(0), 0, 5,
      dimnames=list(NULL, c('path', 'op', 'value', 'code', 'extraValue'))),
      constraintLogic=NULL)
```

Arguments

name	a string, representing the name of the query.
view	a character vector, representing the fields to be selected from InterMine.
sortOrder	a string, representing the field according to which the query result is sorted and the sort order ("asc" or "desc"), following the format "FIELD ORDER".
longDescription	a string, representing the description of the query.
constraints	a matrix of 5 columns (path, op, value, code, extraValue), with each row representing a constraint of the query.
constraintLogic	a string, representing the logical relationship between the constraints, e.g., "A or B" where "A" and "B" are the codes in the constraints.

Value

a list, representing the query.

Author(s)

InterMine Team

Examples

```
nq <-  newQuery()
```

---

renameList *Rename a list stored on the mine*

Description

Rename a list stored on the mine.

Usage

```
renameList(im, old.name, new.name, timeout = 3)
```

Arguments

im	a list, containing the base URL and API token.
old.name	a string, representing the old name of the list.
new.name	a string, representing the new name of the list.
timeout	an integer, representing the number of seconds to wait for the webservice to respond.

Value

a list, representing the response from the server for the rename operation.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta", "TOKEN") #replace TOKEN with your token

## Not run: status <- renameList(im, "a", "b")
```

`runQuery` *Run a query*

Description

Returns results from a query against data held inside the mine. These queries are similar to SQL queries, in that they request certain defined output columns of output, filtering the results through a series of "constraints".

Usage

```
runQuery(im, qry, format="data.frame", timeout = 60)
```

Arguments

`im`	a list, containing the base URL and API token.
`qry`	a list or XML string, representing the query to the database.
`format`	a string, representing the output format of the query result. It could be one of these values: "data.frame", "sequence".
`timeout`	an integer, representing the number of seconds to wait for the webservice to respond.

Value

If format="data.frame", the function returns a data.frame object, representing the query result.

If format="sequence", the function returns an XStringSet object, representing the DNA sequences resulted from the query.

Author(s)

InterMine Team

Examples

```
im <- initInterMine("metabolicmine.org/beta")

queryGeneIden <- getTemplateQuery(im, "Gene_Identifiers")
queryGeneIden$view
queryGeneIden$constraints

qryRes <- runQuery(im, queryGeneIden)
```

Index

deleteList, 2
doEnrichment, 3

getLists, 4
getModel, 5
getRegionFeature, 6
getRegionSequence, 7
getRelease, 7
getTemplateQuery, 8
getTemplates, 9
getVersion, 10

initInterMine, 10

listMines, 11

newList, 12
newQuery, 12

renameList, 13
RInterMine *(*RInterMine-package*)*, 2
RInterMine-package, 2
runQuery, 14