Package ‘AnnotationFuncs’

March 22, 2017

Maintainer Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>
License GPL-2

BugReports
Title Annotation translation functions
Author Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>
Description Functions for handling translating between different
dero Bt.db).
Version 1.24.0
biocViews AnnotationData, Software
Date 2010-11-29
Depends R (>= 2.7.0), AnnotationDbi
Imports DBI
Suggests org.Bt eg.db, GO.db, org.Hs eg.db, hom.Hs.inp.db
Collate ‘annotation-funcs.R’ ‘homologe.R’
NeedsCompilation no

R topics documented:

  AnnotationFuncs-package ........................................ 2
  .dbEscapeString ....................................................... 3
  .getTableName ......................................................... 3
  .pickRef ............................................................. 4
  getEvidenceCodes ..................................................... 5
  getOrthologs .......................................................... 5
  mapLists .............................................................. 7
  pickGO ............................................................... 8
  pickRefSeq ........................................................... 9
  removeNAs ............................................................ 10
  translate ............................................................. 10

Index 13
Description

Package: AnnotationFuncs
Type: Package
Version: 1.3.0
Date: 2011-06-10
License: GPL-2
LazyLoad: yes

Details

Functions for handling translations between different identifiers using the Biocore Data Team data-packages (e.g. org.Bt.eg.db). Primary functions are `translate` for translating and `getOrthologs` for efficient lookup of homologues using the Inparanoid databases. Other functions include functions for selecting Refseqs or Gene Ontologies (GO).

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

References


See Also

`translate`, `getOrthologs`

Examples

```r
library(org.Bt.eg.db)
gene.symbols <- c("DRBP1", "SERPINA1", "FAKE", "BLABLA")
# Find entrez identifiers of these genes.
eg <- translate(gene.symbols, org.Bt.egSYMBOL2EG)
# Note that not all symbols were translated.

# Go directly to Refseq identifiers.
refseq <- translate(gene.symbols, from=org.Bt.egSYMBOL2EG, to=org.Bt.egREFSEQ)
# Pick the proteins:
pickRefSeq(refseq, priorities=c("NP", "XP"), reduce='all')
```
**.dbEscapeString**

---

### Description

Private Escape string

### Usage

```
.dbEscapeString(str, raise.error=TRUE)
```

### Arguments

- **str**: String to test
- **raise.error**: Logical, whether to raise an error or not.

### Details

Does not escape strings, but raises an error if any character expect normal letters and underscores are found in the string.

### Value

Invisible logical

---

**.getTableName**

---

### Description

Gets the table name from the INPARANOID style genus names.

### Usage

```
.getTableName(genus)
```

### Arguments

- **genus**: 5 character INPARANOID genus name, such as "BOSTA", "HOMSA" or "MUSMU".

### Details

The INPARANOID style genus name is a 5 letter acronym of the species name. Quote INPARANOID (?hom.Hs.inpBOSTA):

*Names for these maps are done in the "INPARANOID style" which means that they are normally the 1st three letters of the genus followed by the 1st two letters of the species. For example: "Mus musculus" becomes "MUSMU", "Homo sapiens" becomes "HOMSA", "Monodelphis domestica" becomes "MONDO" etc. This means that for most of these organisms it will be possible to easily guess the abbreviations used. An exception may occur in the future if a new model organism has a very similar genus and species name to an existing one.*
**.pickRef**

**Value**

Table name for genus.

**Author(s)**

Stefan McKinnon Edwards <stefan.m.edwards@agrsci.dk>

**References**

http://www.bioconductor.org/packages/release/bioc/html/AnnotationDbi.html

---

**.pickRef**  
*Secret function that does the magic for pickRefSeq.*

---

**Description**

Secret function that does the magic for pickRefSeq.

**Usage**

```
.pickRef(l, priorities, reduce=c("all", "first", "last"))
```

**Arguments**

- `l` List.
- `priorities` How to prioritize.
- `reduce` How to reduce.

**Details**

Do not use it, use `pickRefSeq`!

**Value**

List.

**Note**

Hey, you found a secret function! Keep it that way!

**Author(s)**

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

**See Also**

pickRefSeq
getEvidenceCodes

Returns GO evidence codes.

Description

Returns GO evidence codes.

Value

Matrix of two columns, first column with codes, second column with description of codes.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

References

?org.Bt.egGO

See Also

pickGO

Examples

gEvidenceCodes()

getOrthologs

Performs quicker lookup for orthologs in homologe data packages...

Description

Performs quicker lookup for orthologs in homologe data packages

Usage

gOrthologs(values, mapping, genus, threshold=1, pre.from, pre.to, 
    post.from, post.to, ...)

Arguments

values
    Vector, coerced to character vector, of values needed mapping by homology.

mapping
    Homology mapping object, such as hom.Hs.inpBOSTA or revmap(hom.Hs.inpBOSTA).

genus
    Character vector. 5 character INPARANOID style genus name of the mapping object, e.g. 'BOSTA' for both hom.Hs.inpBOSTA and revmap(hom.Hs.inpBOSTA).

threshold
    Numeric value between 0 and 1. Only clustered homologues with a pairwise score above the threshold is included. The native implementation has this set to 1.
getOrthologs

pre.from    Mapping object if values needs translation before mapping. E.g. values are entrez and hom.Hs.inpBOSTA requires ENSEMBLPROT, hom.Hs.inpAPIME requires Refseq (?). Arguments from and to are just like in translate.

pre.to    Second part of translation before mapping.

post.from    Translate the result from homology mapping to a desired id; just like in translate.

post.to    Second part of translation after mapping.

...    Additional arguments sent to translate.

Details

Using the INPARANOID data packages such as hom.Hs.inp.db is very, very slow and can take up to 11 min (on this particular developers workstation). This function introduces a new method that can do it in just 20 seconds (on the developers workstation). In addition, it includes options for translating between different identifiers both before and after the mapping.

Value

List. Names of list corresponds to values, except those that could not be mapped nor translated. Entries are character vectors.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

References


See Also

translate, .getTableName, mapLists

Examples

library(hom.Hs.inp.db)
library(org.Hs.eg.db)
library(org.Bt.eg.db)
getOrthologs(’ENSBTP00000024572’, revmap(hom.Hs.inpBOSTA), ’BOSTA’)  # And now, we will map from entrez genes 1, 2 and 3 to bovine Refseq
bovine.ensembl <- getOrthologs(c(1,2,3), hom.Hs.inpBOSTA, ’BOSTA’, pre.from=org.Hs.egENSEMBLPROT, post.from=org.Bt.egENSEMBLPROT)
refseqs <- translate(unlist(bovine.ensembl, use.names=FALSE), org.Bt.egREFSEQ)
hs2bt.refseqs <- mapLists(bovine.ensembl, refseqs)  # Another way of doing it:
hs2bt.refseqs2 <- lapply(bovine.ensembl, translate, from=org.Bt.egREFSEQ, simplify=TRUE) # simplify=TRUE is very important here!
mapLists

Replaces contents of list A with elements of list B...

Description
Replaces contents of list A with elements of list B

Usage
mapLists(A, B, removeNAs=TRUE)

Arguments
A List, elements are coerced to character for mapping to B.
B List.
removeNAs Boolean, whether to remove the NAs that occur because an element was not found in B.

Details
Combines two lists, A and B, such that names(A) are preserved, mapping to the values of B, using names(B) as look up. Ie. replaces values in A with values in B, using names(B) as look up for values in A. Once more? See examples. NB! None-mapped entries are returned as NA, but can be removed using removeNAs.

Value
List.

Author(s)
Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

See Also
removeNAs

Examples
A <- list('a1'='alpha','a2'='beta','a3'=c('gamma','delta'))
B <- list('alpha'=c('b1'), 'gamma'=c('b2', 'b3'), 'delta'=c('b4'))
mapLists(A, B)
pickGO  

Cleans up result from org.Xx.egGO and returns specific GO identifiers

Usage

pickGO(l, evidence=NA, category=NA)

Arguments

l  Character vector, or list of, og GO identifiers.

evidence  Character vector, filters on which kind of evidence to return; for a larger list see getEvidenceCodes. * Evidence codes may be: c("IMP", "IGI", "IPI", "ISS", "IDA", "IEP", "IEA")  

* Leave as NA to ignore filtering on this part.

category  Character vector, filters on which ontology to return: biological process (BP), cellular component (CC), or molecular function (MF). * Leave as NA to ignore filtering on this part.

Details

Cleans up result from org.Xx.egGO and returns GO identifier for either biological process (BP), cellular component (CC), or molecular function (MF). Can be used on list of GOs from translate, or a single list of GOs from an annotation package. May reduce list, if the (sub)list does not contain the chosen class!

Value

List with only the picked elements.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

See Also

pickRefSeq, getEvidenceCodes, translate

Examples

library(org.Bt.eg.db)
genes <- c(280705, 280706, 100327208)
GO <- translate(genes, org.Bt.egGO)
# Get all biological processes:
pickGO(GO, category="BP")
# Get all ontologies with experimental evidence:
pickGO(GO, evidence=c("IMP", "IGI", "IPI", "ISS", "IDA", "IEP", "IEA"))
**pickRefSeq**  
*Picks a prioritised RefSeq identifier from a list of identifiers...*

**Description**  
Picks a prioritised RefSeq identifier from a list of identifiers

**Usage**  
```
pickRefSeq(l, priorities=c("NP", "XP", "NM", "XM"), reduce=c("all", "first", "last"))
pickRefSeq.mRNA(l)
pickRefSeq.Protein(l)
```

**Arguments**  
- **l**: Vector or list of RefSeqs accessions to pick from. If list given, applies the prioritization to each element in the list.
- **priorities**: Character vector of prioritised prefixes to pick by. Eg. c("NP", "NM") returns RefSeqs starting 'NP', and if none found, those starting 'NM'. If no RefSeqs are found according to the priorities, Null is returned, unless the last element in priorities is '*'. Uses grepl, so see these for pattern matching. Default: c('NP','XP','NM','XM')
- **reduce**: Reducing method, either return all annotations (one-to-many relation) or the first or last found annotation. The reducing step is applied after translating to the goal: all: returns all annotations first or last: choose first or last of arbitrarily ordered list.

**Details**  
When translating to RefSeq, typically multiple identifiers are returned, referring to different types of products, such as genomic molecule, mature mRNA or the protein, and they can be predicted, properties that can be read from the prefix [http://www.ncbi.nlm.nih.gov/refseq/key.html](http://www.ncbi.nlm.nih.gov/refseq/key.html). E.g. "XM_" is predicted mRNA and "NP_" is a protein. Run ?org.Bt.eGREFSEQ.

**Value**  
If vector given, returns vector. If list given, returns list without element where nothing could be picked.

**Author(s)**  
Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

**Examples**  
```
library(org.Bt.eG.db)
symbols <- c("SERPINA", "KERA", "CDS")
refseq <- translate(symbols, from=org.Bt.eGSYMBOL2EG, to=org.Bt.eGREFSEQ)
mRNA <- pickRefSeq(refseq, priorities=c("NM","XM"))
proteins <- pickRefSeq(refseq, priorities=c("NP","XP"))
# The same.
```
mRNA <- pickRefSeq.mRNA(refseq)
proteins <- pickRefSeq.Protein(refseq)

removeNAs

Removes entries equal NA from list or vector...

Description
Removes entries equal NA from list or vector

Usage
removeNAs(l)

Arguments
l Vector or list.

Details
Removes entries equal NA, but not mixed entries containing, amongst others, NA. Good for use after mapLists that might return entries equal NA.

Author(s)
Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

Examples
removeNAs(list('a'=NA, 'b'=c(NA, 'B'), 'c'='C'))

translate

Translate between different identifiers...

Description
Translate between different identifiers

Usage
translate(values, from, to, reduce=c("all", "first", "last"),
return.list=TRUE, remove.missing=TRUE, simplify=FALSE, ...)
translate

Arguments

values  Vector of annotations that needs translation. Coerced to character vector.

from  Type of annotation values are given in. NB! take care in the orientation of the package, ie. if you have RefSeq annotations, use org.Bt.egREFSEQ2EG or (in some cases) revmap(org.Bt.egREFSEQ).

to  Desired goal, eg. org.Bt.egENSEMBLPROT. If NULL (default), goal if the packages primary annotation (eg. entrez gene for org.Bt.eg.db). Throws a warning if the organisms in from and to are not the same.

reduce  Reducing method, either return all annotations (one-to-many relation) or the first or last found annotation. The reducing step is applied after translating to the goal: all: returns all annotations first or last: choose first or last of arbitrarily ordered list.

return.list  Logical, when TRUE, returns the translation as a list where names

remove.missing  Logical, whether to remove non-translated values, defaults TRUE.

simplify  Logical, unlists the result. Defaults to FALSE. Usefull when using translate in a lapply or sapply.

...  Additional arguments sent to pickGO if from returns GO set.

Details

Function for translating from one annotation to another, eg. from RefSeq to Ensemble. This function takes a vector of annotation values and translates first to the primary annotation in the Biocore Data Team package (ie. entrez gene identifier for org.Bt.eg.db) and then to the desired product, while removing non-translated annotations and optionally reducing the result so there is only a one-to-one relation.

If you want to do some further mapping on the result, you will have to use either unlist og 1apply, where the first returns all the end-products of the first mapping, returning a new list, and the latter produces a list-within-list.

If from returns GO identifiers (e.g. from = org.Bt.egGO), then the returned resultset is more complex and consists of several layers of lists instead of the usual list of character vectors. If to has also been specified, the GO IDs must be extracted (internally) and you have the option of filtering for evidence and category at this point. See pickGO.

Value

List; names of elements are values and the elements are the translated elements, or NULL if not translatable with remove.missing = TRUE.

Note

Requires user to deliver the annotation packages such as org.Bt.egREFSEQ.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

See Also

pickRefSeq, pickGO
Examples

```r
library(org.Bt.eg.db)
genes <- c(280705, 280706, 100327208)
translate(genes, org.Bt.egSYMBOL)

symbols <- c("SERPINA1","KERA","CD5")
refseq <- translate(symbols, from=org.Bt.egSYMBOL2EG, to=org.Bt.egREFSEQ)
# Pick the proteins:
pickRefSeq(refseq, priorities=c("NP","XP"), reduce='all')

# If you wanted do do some further mapping on the result from
translate, simply use lapply.

library(GO.db)
GO <- translate(genes, org.Bt.egGO)
# Get all biological processes:
pickGO(GO, category="BP")
# Get all ontologies with experimental evidence:
pickGO(GO, evidence=c("IMP","IGI","IPI","ISS","IDA","IEP","IEA"))
```
Index

«Topic package
   AnnotationFuncs-package, 2
   .dbEscapeString, 3
   .getTableName, 3, 6
   .pickRef, 4

AnnotationFuncs
   (AnnotationFuncs-package), 2
AnnotationFuncs-package, 2

getEvidenceCodes, 5, 8
getOrthologs, 2, 5
mapLists, 6, 7, 10

pickGO, 5, 8, 11
pickRefSeq, 4, 8, 9, 11

removeNAs, 7, 10

translate, 2, 6, 8, 10