KEGGSOAP

November 11, 2009

R topics documented:

`bget` *Client-side interface to obtain KEGG database entries by a list of entry identifiers*

Description

`bget` is used for retrieving KEGG database entries specified by a list of entry identifiers. It accepts all the KEGG bget commond line options as a character string. Number of entries retrieved at a time is restricted up to 100.

Usage

```
bget(bget.command)
```

Arguments

`bget.command` `bget.command` a character string of KEGG bget command

Value

a character string of KEGG bget search result.

Author(s)

Nianhua Li

References

http://www.genome.jp/kegg/docs/keggapi_manual.html#label:40

Examples

```
# retrieve two KEGG/GENES entries
bget("eco:b0002 hin:tRNA-Cys-1")
# retrieve nucleic acid sequences in a FASTA format
bget("-f -n n eco:b0002 hin:tRNA-Cys-1")
# retrieve amino acid sequence in a FASTA format
bget("-f -n a eco:b0002")
```

getBestNeighbors *Client-side interface to obtain the name of genes that are homologous to a given gene*

Description

Given a KEGG gene id, the functions query the KEGG Sequence Similarity Database (SSDB) for genes that are homologous to the target gene in other organisms. Genes that share an arbitrary threshold level of similarity determined by alignment of matching bases are termed homologous.

Usage

```
get.best.best.neighbors.by.gene(genes.id, start, max.results)
get.best.neighbors.by.gene(genes.id, start, max.results)
getBestNeighbors(genes.id, start, max.results, what = c("best", "best_best"))
```

Arguments

genes.id	genes.id a character string for the id used by KEGG to represent the gene of interest. The id normally consists of three letters followed by a colon and then several numbers. The three letters are from the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern (e. g. hsa:111 for Homo Sapiens)
start	start an integer to indicate the location of the entry in the query results from which the results will be extracted and returned
max.results	max.results an integer to indicate the maximum number of entries that will be extracted from the query results and returned
what	what a character string that can either be "best" or "best_best" to indicate whether reciprocal homologous genes are sought

Details

A given gene may have several homologous genes across organisms. A query to SSDB will have a list of genes that are homologous to the target gene. `start` and `max.results` indicate where on the list to start and stop to extract data and return the results.

`getBestNeighbors` is a general function that queries the SSDB database and gets the results based on whether the query is for best or best best homologous relationships.

Value

The functions return a list of lists. Each sub-list contains data for a gene that is homologous to the target gene with the following elements:

`genes_id1` a character string for the id of the target gene used to query for hologous genes

`genes_id2` a character string for the id of the homologous gene found in another organism

`sw_score` an integer for Smith-Waterman score between genes_id1 and genes_id2

`bit_score` a numeric value for the bit score between genes_id1 and genes_id2

`identity` a numeric value between 0 and 1 for the degree of identity between genes_id1 and genes_id2

`overlap` an integer for the overlapping length between genes_id1 and genes_id2

`start_position1` an integer for the start position of the alignment in genes_id1

`end_position1` an integer for the end position of the alignment in genes_id1

`start_position2` an integer for the start position of the alignment in genes_id2

`end_position2` an integer for the end position of the alignment in genes_id2

`best_flag_1to2` a boolean that is TRUE if genes_id2 is the best neighbor gene of genes_id1

`best_flag_2to1` a boolean that is TRUE if genes_id1 is also the best neighbor gene of genes_id2

`definition1` a character string for the definition of genes_id1

`definition2` a character string for the definition of genes_id2

`length1` an integer for the amino acid length of the genes_id1

`length2` an integer for the amino acid length of the genes_id2

Author(s)

Jianhua Zhang

References

`http://www.genome.jp/kegg/soap/doc/keggapi_manual.html`

See Also

`get.genes.by.organism`

Examples

```
        bestGenes <- get.best.neighbors.by.gene("eco:b0002",1, 5)
        bestBestGenes <- get.best.best.neighbors.by.gene("eco:b0002",1, 5)
```

---

get.genes.by.motifs

Client-side interface to obtain the name of genes that contain the motifs represented by a set of motif ids

---

Description

Given a set of motif ids, the function searches the databases implied by the motif ids for genes containing the motifs specified by the motif ids.

Usage

```
get.genes.by.motifs(motif.id.list, start, max.results)
```

Arguments

`motif.id.list`
: `motif.id.list` a vector of character strings for the ids of the motifs that are conserved by genes across organisms

`start`
: `start`an integer to indicate the location of the entry in the query results from which the results will be extracted and returned

`max.results`
: `max.results` an integer to indicate the maximum number of entries that will be extracted from the query results and returned

Details

KEGG seems to have two ways of defining the ids for motifs. One is the motif ids obtained through `get.motifs.by.gene`, where pfam, tfam, pspt, pspf are used for the Pfam, TIGRFAM, PROSITE pattern, and PROSITE profile database, respectively and for the first part of a motif id (e. g. pfam:aakinase). Another is the motif ids used to query the databases for genes that contain the motif, where only the first two letters of the abbreviations for databases form the first part of a motif id (e. g. pf:aakinase)

Value

The function returns a named vector with the names of the vector being the textual definition of genes and values of the vector being the ids used by KEGG to represent genes

Author(s)

Jianhua Zhang

References

`http://www.genome.jp/kegg/soap/doc/keggapi_manual.html`

See Also

`get.motifs.by.gene`

Examples

```
        genes <- get.genes.by.motifs(c("pf:DnaJ", "ps:DNAJ_2"), 1, 10)
```

---

get.genes.by.organism

Client-side interface to obtain the KEGG ids for all the genes of a given organism

---

Description

Given a KEGG organism id, the function searches the KEGG GENES database for all the genes of the organism

Usage

```
get.genes.by.organism(org, start, max.results)
```

Arguments

`org`	`org` a character string for the id used by KEGG for organisms. The organism ids are normally three-letter codes with the first letter being the first letter of the genus name and the rest being the first two letters of the species name of the scientic name of the organism of concern
`start`	`start` an integer to indicate the location of the entry in the query results from which the results will be extracted and returned
`max.results`	`max.results` an integer to indicate the maximum number of entries that will be extracted from the query results and returned

Details

The gene ids returned by the query normally consist of three letters followed by a colon and then numbers or a combination of letters and numbers. The three letters are from the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern (e. g. hsa:111 for Homo Sapiens)

Value

The function returns a vector of character strings of ids used by KEGG to represent genes

Author(s)

Jianhua Zhang

References

`http://www.genome.jp/kegg/soap/doc/keggapi_manual.html`

Examples

```
          genes <- get.genes.by.organism("hsa", 1, 10)
```

`get.genes.by.pathway` *Client-side interface to obtain the KEGG ids for genes/enzymes/compounds/reactions that are involved in the interactions in a given pathway*

Description

Given a KEGG pathway identifier, the functions query the KEGG PATHWAY database for all the genes/enzymes/compounds/reactions that that are involved in the interactions in the specified pathway.

Usage

```
get.genes.by.pathway(pathway.id)
get.enzymes.by.pathway(pathway.id)
get.compounds.by.pathway(pathway.id)
get.reactions.by.pathway(pathway.id)
```

Arguments

`pathway.id`	`pathway.id` a character string for a KEGG pathway id. KEGG pathway ids consist of the string path followed by a colon, a three-letter code for the organism of concern, and then a number (e. g. "path:eco00020"). The three-letter organism code consists of the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern

Details

KEGG pathway identifiers for a given organism can be obtained using function `list.pathways`

Value

The functions return a vector of KEGG gene/enzyme/compound/reation ids found in the pathway

Author(s)

Jianhua Zhang

References

`http://www.genome.jp/kegg/soap/doc/keggapi_manual.html`

See Also

`list.pathways`

Examples

```
genes <- get.genes.by.pathway("path:eco00020")
enzymes <- get.enzymes.by.pathway("path:eco00020")
compounds <- get.compounds.by.pathway("path:eco00020")
reactions <- get.reactions.by.pathway("path:eco00020")
```

`get.motifs.by.gene` *Client-side interface to obtain the name of genes that are homologous to a given gene*

Description

This function queries the Pfam, TIGRFAM, PROSITE pattern, and/or PROSITE profile databases for the motifs of a given gene. A motif is a locally conserved region of a sequence or a short sequence pattern shared by a set of sequences

Usage

```
get.motifs.by.gene(genes.id, db)
```

Arguments

`genes.id` `genes.id` a character string for the id used by KEGG to represent the gene of interest. The id normally consists of three letters followed by a colon and then several numbers. The three letters are from the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern (e. g. hsa:111 for Homo Sapiens)

`db` `db` a character string for the name of the data to search for motifs. Valid database names include pfam, tfam, pspt, pspf for the Pfam, TIGRFAM, PROSITE pattern, and PROSITE profile, respectively, or all for all the four databases

Details

The motif ids obtained can be used to search for the genes that contain the motif across organism using `get.genes.by.motifs`

Value

The function returns a list of lists with each of the sub-list having the following elements:

`motif_id` a character string for the id of the motif found

`definition` a character string for the definition of the motif

`genes_id` a character string for the KEGG genes_id of the gene that contains the motif and used to search the database(s)

`start_position`
an integer for the start position of the motif match

`end.position` an integer for the end position of the motif match

`score` a numeric value for the score of the motif match for TIGRFAM and PROSITE databases

`evalue` a numeric value for the E-value of the motif match for Pfam database

Author(s)

Jianhua Zhang

References

http://www.genome.jp/kegg/soap/doc/keggapi_manual.html

See Also

`get.genes.by.motifs`

Examples

```
motifs <- get.motifs.by.gene("eco:b0002", "pfam")
sapply(motifs, function(x) x$motif_id)
```

---

get.paralogs.by.gene

Client-side interface to obtain data for paralogous genes

Description

Given a KEGG gene id, the function queries the KEGG Sequence Similarity Database (SSDB) for genes that are paralogous to the target gene. Paralogous genes result from duplication of existing genes and then function divergence

Usage

```
get.paralogs.by.gene(genes.id, start, max.results)
```

Arguments

`genes.id`	`genes.id` a character string for the id used by KEGG to represent the gene of interest. The id normally consists of three letters followed by a colon and then several numbers. The three letters are from the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern (e. g. hsa:111 for Homo Sapiens)
`start`	`start` an integer to indicate the location of the entry in the query results from which the results will be extracted and returned
`max.results`	`max.results` an integer to indicate the maximum number of entries that will be extracted from the query results and returned

Details

A given gene may have several paralogous genes. A query to SSDB may have a list of genes that are paralogous to the target gene. `start` and `max.results` indicate where on the list to start and stop to extract data and return the results.

Value

The function returns a list of lists. Each sub-list contains data for a gene that is paralogous to the target gene with the following elements:

`genes_id1` a character string for the id of the target gene used to query for hologous genes

`genes_id2` a character string for the id of the homologous gene found in another organism

`sw_score` an integer for Smith-Waterman score between genes_id1 and genes_id2

`bit_score` a numeric value for the bit score between genes_id1 and genes_id2

`identity` a numeric value between 0 and 1 for the degree of identity between genes_id1 and genes_id2

`overlap` an integer for the overlapping length between genes_id1 and genes_id2

`start_position1`
an integer for the start position of the alignment in genes_id1

`end_position1`
an integer for the end position of the alignment in genes_id1

`start_position2`
an integer for the start position of the alignment in genes_id2

`end_position2`
an integer for the end position of the alignment in genes_id2

`best_flag_1to2`
a boolean that is TRUE if genes_id2 is the best neighbor gene of genes_id1

`best_flag_2to1`
a boolean that is TRUE if genes_id1 is also the best neighbor gene of genes_id2

`definition1` a character string for the definition of genes_id1

`definition2` a character string for the definition of genes_id2

`length1` an integer for the amino acid length of the genes_id1

`length2` an integer for the amino acid length of the genes_id2

Author(s)

Jianhua Zhang

References

`http://www.genome.jp/kegg/soap/doc/keggapi_manual.html`

See Also

`get.best.neighbors.by.gene`

Examples

```
paraGenes <- get.paralogs.by.gene("eco:b0002", 1, 10)
```

`get.pathways.by.genes` *Client-side interface to obtain the KEGG pathway ids*

Description

Given a set of KEGG gene/enzyme/compound/reation identifiers, the functions query the KEGG PATHWAY database for all the pathways in which items represented by the given set of identifers are involved

Usage

```
get.pathways.by.genes(genes.id.list)
get.pathways.by.enzymes(enzyme.id.list)
get.pathways.by.compounds(compound.id.list)
get.pathways.by.reactions(reaction.id.list)
```

Arguments

`genes.id.list`
: `genes.id.list` a vector of character strings for the ids used by KEGG to represent genes. An id normally consists of three letters followed by a colon and then several numbers. The three letters are from the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern (e. g. hsa:111 for Homo Sapiens)

`enzyme.id.list`
: `enzyme.id.list` a vector of character strings for enzyme commission numbers

`compound.id.list`
: `compound.id.list` a vector of character strings for the ids used by KEGG to represent compounds. A compound id begins with cpd: followed by a combination of letters and numbers (e. g. cpd:C00579)

`reaction.id.list`
: `reaction.id.list` a vector of character strings for the ids used by KEGG to represent reactions. A reaction id begins with rn: followed by a combination of letters and numbers (e. g. rn:R00268)

Value

The functions return a vector of KEGG pathway ids

Author(s)

Jianhua Zhang

References

http://www.genome.jp/kegg/soap/doc/keggapi_manual.html

See Also

get.genes.by.pathway, get.enzymes.by.pathway, get.compounds.by.pathway, get.reactions.by.pathway

Examples

```
# There seems to be some problem at the server side. Use try
pathways <- try(get.pathways.by.genes(c("eco:b0077", "eco:b0078")))
pathways <- try(get.pathways.by.enzymes("ec:1.3.99.1"))
pathways <- try(get.pathways.by.compounds(c("cpd:C00033", "cpd:C00158")))
pathways <- try(get.pathways.by.reactions(c("rn:R00959",
                         "rn:R02740", "rn:R00960", "rn:R01786")))
```

KEGGserver *Definitions of objects used by other functions to access KEGG SOAP service*

Description

Definitions of KEGG SOAP server, KEGG SOAP action, and KEGG XML name space are made when the package is invoked so that they are available to other functions

Details

All the functions that envoke KEGG SOAP services have a KEGG server, KEGG action, and KEGG XML name space as argument. These objects are defined in .First.lib and made available to the functions when the package is invoked

Author(s)

Jianhua Zhang

References

http://www.genome.jp/kegg/soap/doc/keggapi_manual.html

---

list.organisms *Client-side interface to obtain the names of organisms supported by KEGG databases*

Description

These functions provides an R interface to allow users to get the names/ids of organisms, databases, pathways that are available through KEGG SOAP services.

Usage

```
list.organisms()
list.pathways(org)
list.databases()
```

Arguments

`org`	`org` a character string for the id used by KEGG for organisms. The organism ids are normally three-letter codes with the first letter being the first letter of the genus name and the rest being the first two letters of the species name of the scientic name of the organism of concern

Details

Some queries agsinst the KEGG databases require abbreviations of organisms supported by KEGG. Although the abbreviations normally consist of three letters by truncating the first letter of the genus name and the first two letters of the species name (e. g. hsp for Homo sapiens), `list.organisms` obtains the abbreviations using the service provided by KEGG SOAP to make sure the abbreviations are correct and the organisms are indeed supported by KEGG databases.

Value

`list.organisms` returns a named vector with names of the vector being the scientific names and the values of the vector being the abbreviations used by KEGG for the organisms supported by the databases.

`list.pathways` returns a named vector with names of the vector being textual descriptions of KEGG pathways and the values of the vector being the ids used by KEGG to represent pathways.

`list.databases` returns

Author(s)

Jianhua Zhang

References

http://www.genome.jp/kegg/soap/doc/keggapi_manual.html

Examples

```
        list.organisms()
```

`mark.pathway.by.objects` *Client-side interface to obtain an url for a KEGG pathway diagram with a given set of genes marked*

Description

Given a KEGG pathway id and a set of KEGG gene ids, the functions return the URL of a KEGG pathway diagram with the elements corresponding to the genes marked by red or specified color

Usage

```
mark.pathway.by.objects(pathway.id, object.id.list)
color.pathway.by.objects(pathway.id, object.id.list,
                                     fg.color.list, bg.color.list)
```

Arguments

`pathway.id` `pathway.id` a character string for a KEGG pathway id. KEGG pathway ids consist of the string path followed by a colon, a three-letter code for the organism of concern, and then a number (e. g. "path:eco00020"). The three-letter organism code consists of the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern

`object.id.list` `object.id.list` a vector of character strings for KEGG gene ids. KEGG gene ids normally consist of three letters followed by a column and then several numeric numbers. The three letters are from the first letter of the genus name and the first two letters of the species name of the scientific name of the organism of concern (e. g. hsa:111 for Homo Sapiens)

`fg.color.list` `fg.color.list` a vector of two character strings to indicate the color for the text and border, respectively, of the objects in a pathway diagram. The strings can either be a color code linke #ff0000 or letter link yellow

`bg.color.list` `bg.color.list` a vector of character strings of the same length of `object.id.list` to indicate the background color of the objects in a pathway diagram. The strings can either be a color code like #ff0000 or letter like yellow

Details

This function only returns the URL of the KEGG pathway diagram. Use the function `browseURL` to view the diagram

Value

This function returns a character string for the url

Author(s)

Jianhua Zhang

References

`http://www.genome.jp/kegg/soap/doc/keggapi_manual.html`

See Also

`browseURL`

Examples

```
 url <- mark.pathway.by.objects("path:eco00260",
               c("eco:b0002", "eco:c00263"))
 if(interactive()){
     browseURL(url)
 }
url <- color.pathway.by.objects("path:eco00260",
               c("eco:b0002", "eco:c00263"),
               c("#ff0000", "#00ff00"), c("#ffff00", "yellow"))
```

search.compounds.by.name *Client-side interface to obtain a list of chemical compounds*

Description

The functions provide access to KEGG LIGAND database http://www.genome.jp/kegg/ligand.html. Given a compound name, a chemical formula, a molecular weight, or a common sub-structure, one of the functions below can return a list of compounds identifiers from KEGG LIGAND database.

Usage

```
search.compounds.by.name(name)
search.compounds.by.composition(composition)
search.compounds.by.mass(mass, range)
search.compounds.by.subcomp(mol, offset, limit)
```

Arguments

name	name a character string to indicate a compound name
composition	composition a character string to indicate a compound composition, usually expressed as chemical formula
mass	massa float to indicate a molecular weight around mass
range	range a float to indicate the range of molecular weight when searching compounds by mass
mol	mola character string to indicate a MOL formatted structural data, more in details section
offset	offset an integer
limit	limit an integer

Details

search.compounds.by.name returns a list of compounds having the specified name;

search.compounds.by.composition returns a list of compounds containing elements indicated by the composition. Order of the elements is insensitive;

search.compounds.by.mass returns a list of compounds having the molecular weight around "mass" with some ambiguity (range);

search.compounds.by.subcomp returns a list of compounds with the alignment having common sub-structure calculated by the subcomp program. You can obtain a MOL formatted structural data of matched compounds using bget with the "-f m" option to confirm the alignment.

Value

All the functions return a character vector of chemical compound identifiers provided by KEGG LIGAND database

Author(s)

Nianhua Li

References

http://www.genome.jp/kegg/docs/keggapi_manual.html#label:105

See Also

bget

Examples

```
compounds_1 <- search.compounds.by.name("shikimic acid")
compounds_2 <- search.compounds.by.composition("C7H10O5")
compounds_3 <- search.compounds.by.mass(174.05, 0.1)
mol <- bget("-f m cpd:C00111")
compounds_4 <- search.compounds.by.subcomp(mol, 1, 5)
```

---

search.glycans.by.name

Client-side interface to obtain a list of chemical glycans

Description

The functions provide access to KEGG LIGAND database http://www.genome.jp/kegg/ligand.html. Given a glycan name, a composition, a molecular weight, or a common substructure, one of the functions below can return a list of glycans identifiers from KEGG LIGAND database.

Usage

```
search.glycans.by.name(name)
search.glycans.by.composition(composition)
search.glycans.by.mass(mass, range)
search.glycans.by.kcam(kcf, program, option, offset, limit)
```

Arguments

name	name a character string to indicate a glycan name
composition	composition a character string to indicate the composition of monosaccharides
mass	mass a float to indicate the mass computed from the composition, excluding those in parentheses
range	range a float to indicate the range of molecular weight when searching glycans by mass
kcf	kcf a character string to indicate the molecular structure (carbohydrate sequence) of a glycan in KCF format
program	program a character string, either "gapped" or "ungaped"
option	option a character string, either "global" or "local"
offset	offset an integer
limit	limit an integer

Details

search.glycans.by.name returns a list of glycans having the specified name;

search.glycans.by.composition returns a list of glycans containing sugars indicated by the composition. Order of the sugars (in parenthesis with number) is insensitive;

search.glycans.by.mass returns a list of glycans having the molecular weight around "mass" with some ambiguity (range);

search.glycans.by.subcomp returns a list of glycans with the alignment having common sub-structure calculated by the KCaM program. You can obtain a KCF formatted structural data of matched glycans using bget with the "-f m" option to confirm the alignment.

Value

All the functions return a character vector of glycan identifiers provided by KEGG LIGAND database

Author(s)

Nianhua Li

References

http://www.genome.jp/kegg/docs/keggapi_manual.html#label:105

See Also

bget

Examples

```
glycans_1 <- search.glycans.by.name("Paragloboside")
glycans_2 <- search.glycans.by.composition("(Man)4 (GalNAc)1")
glycans_3 <- search.glycans.by.mass(689.6, 0.1)
kcf <- bget("-f k gl:G12922")
glycans_4 <- search.glycans.by.kcam(kcf, "gapped", "local", 1, 5)
```

Index