chicken.db

September 24, 2013

---

chickenACCNUM *Map Manufacturer identifiers to Accession Numbers*

Description

chickenACCNUM is an R object that contains mappings between a manufacturer's identifiers and manufacturers accessions.

Details

For chip packages such as this, the ACCNUM mapping comes directly from the manufacturer. This is different from other mappings which are mapped onto the probes via an Entrez Gene identifier.

Each manufacturer identifier maps to a vector containing a GenBank accession number.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

Examples

```
x <- chickenACCNUM
# Get the probe identifiers that are mapped to an ACCNUM
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the ACCNUM for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

chickenALIAS2PROBE *Map between Common Gene Symbol Identifiers and Manufacturer Identifiers*

Description

chickenALIAS is an R object that provides mappings between common gene symbol identifiers and manufacturer identifiers.

Details

Each gene symbol is mapped to a named vector of manufacturer identifiers. The name represents the gene symbol and the vector contains all manufacturer identifiers that are found for that symbol. An NA is reported for any gene symbol that cannot be mapped to any manufacturer identifiers.

This mapping includes ALL gene symbols including those which are already listed in the SYMBOL map. The SYMBOL map is meant to only list official gene symbols, while the ALIAS maps are meant to store all used symbols.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

Examples

```
# Convert the object to a list
xx <- as.list(chickenALIAS2PROBE)
if(length(xx) > 0){
    # Get the probe identifiers for the first two aliases
    xx[1:2]
    # Get the first one
    xx[[1]]
}
```

chicken.db *Bioconductor annotation data package*

Description

Welcome to the chicken.db annotation Package. The purpose of this package is to provide detailed information about the chicken platform. This package is updated biannually.

You can learn what objects this package supports with the following command:

`ls("package:chicken.db")`

Each of these objects has their own manual page detailing where relevant data was obtained along with some examples of how to use it.

Examples

```
ls("package:chicken.db")
```

---

chickenCHR *Map Manufacturer IDs to Chromosomes*

Description

chickenCHR is an R object that provides mappings between a manufacturer identifier and the chromosome that contains the gene of interest.

Details

Each manufacturer identifier maps to a vector of chromosomes. Due to inconsistencies that may exist at the time the object was built, the vector may contain more than one chromosome (e.g., the identifier may map to more than one chromosome). If the chromosomal location is unknown, the vector will contain an NA.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

Examples

```
x <- chickenCHR
# Get the probe identifiers that are mapped to a chromosome
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the CHR for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

---

chickenCHRLENGTHS *A named vector for the length of each of the chromosomes*

Description

chickenCHRLENGTHS provides the length measured in base pairs for each of the chromosomes.

Details

This is a named vector with chromosome numbers as the names and the corresponding lengths for chromosomes as the values.

Total lengths of chromosomes were derived by calculating the number of base pairs on the sequence string for each chromosome.

Examples

```
tt <- chickenCHRLENGTHS
# Length of chromosome 1
tt["1"]
```

---

chickenCHRLOC *Map Manufacturer IDs to Chromosomal Location*

Description

chickenCHRLOC is an R object that maps manufacturer identifiers to the starting position of the gene. The position of a gene is measured as the number of base pairs.

The CHRLOCEND mapping is the same as the CHRLOC mapping except that it specifies the ending base of a gene instead of the start.

Details

Each manufacturer identifier maps to a named vector of chromosomal locations, where the name indicates the chromosome. Due to inconsistencies that may exist at the time the object was built, these vectors may contain more than one chromosome and/or location. If the chromosomal location is unknown, the vector will contain an NA.

Chromosomal locations on both the sense and antisense strands are measured as the number of base pairs from the p (5' end of the sense strand) to q (3' end of the sense strand) arms. Chromosomal locations on the antisense strand have a leading "-" sign (e. g. -1234567).

Since some genes have multiple start sites, this field can map to multiple locations.

Mappings were based on data provided by: UCSC Genome Bioinformatics (Gallus gallus) ftp://hgdownload.cse.ucsc.edu/gol
With a date stamp from the source of: 2012-Jun11

Examples

```
x <- chickenCHRLOC
# Get the probe identifiers that are mapped to chromosome locations
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the CHRLOC for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

chickenENSEMBL *Map Ensembl gene accession numbers with Entrez Gene identifiers*

Description

chickenENSEMBL is an R object that contains mappings between manufacturer identifiers and Ensembl gene accession numbers.

Details

This object is a simple mapping of manufacturer identifiers to Ensembl gene Accession Numbers.

Mappings were based on data provided by BOTH of these sources: http://www.ensembl.org/biomart/martview/ ftp://ftp.ncbi.nlm.nih.gov/gene/DATA

For most species, this mapping is a combination of manufacturer to ensembl IDs from BOTH NCBI and ensembl. Users who wish to only use mappings from NCBI are encouraged to see the ncbi2ensembl table in the appropriate organism package. Users who wish to only use mappings from ensembl are encouraged to see the ensembl2ncbi table which is also found in the appropriate organism packages. These mappings are based upon the ensembl table which is contains data from BOTH of these sources in an effort to maximize the chances that you will find a match.

For worms and flies however, this mapping is based only on sources from ensembl, as these organisms do not have ensembl to entrez gene mapping data at NCBI.

Examples

```
x <- chickenENSEMBL
# Get the entrez gene IDs that are mapped to an Ensembl ID
mapped_genes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_genes])
if(length(xx) > 0) {
  # Get the Ensembl gene IDs for the first five genes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
#For the reverse map ENSEMBL2PROBE:
# Convert to a list
xx <- as.list(chickenENSEMBL2PROBE)
if(length(xx) > 0){
   # Gets the entrez gene IDs for the first five Ensembl IDs
   xx[1:5]
   # Get the first one
   xx[[1]]
}
```

chickenENTREZID *Map between Manufacturer Identifiers and Entrez Gene*

Description

chickenENTREZID is an R object that provides mappings between manufacturer identifiers and Entrez Gene identifiers.

Details

Each manufacturer identifier is mapped to a vector of Entrez Gene identifiers. An NA is assigned to those manufacturer identifiers that can not be mapped to an Entrez Gene identifier at this time.

If a given manufacturer identifier can be mapped to different Entrez Gene identifiers from various sources, we attempt to select the common identifiers. If a concensus cannot be determined, we select the smallest identifier.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

References

http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene

Examples

```
x <- chickenENTREZID
# Get the probe identifiers that are mapped to an ENTREZ Gene ID
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the ENTREZID for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

chickenENZYME *Maps between Manufacturer IDs and Enzyme Commission (EC) Numbers*

Description

chickenENZYME is an R object that provides mappings between manufacturer identifiers and EC numbers. chickenENZYME2PROBE is an R object that maps Enzyme Commission (EC) numbers to manufacturer identifiers.

Details

When the chickenENZYME maping viewed as a list, each manufacturer identifier maps to a named vector containing the EC number that corresponds to the enzyme produced by that gene. The names corresponds to the manufacturer identifiers. If this information is unknown, the vector will contain an NA.

For the chickenENZYME2PROBE, each EC number maps to a named vector containing all of the manufacturer identifiers that correspond to the gene that produces that enzyme. The name of the vector corresponds to the EC number.

Enzyme Commission numbers are assigned by the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology http://www.chem.qmw.ac.uk/iubmb/enzyme/ to allow enzymes to be identified.

An Enzyme Commission number is of the format EC x.y.z.w, where x, y, z, and w are numeric numbers. In chickenENZYME2PROBE, EC is dropped from the Enzyme Commission numbers.

Enzyme Commission numbers have corresponding names that describe the functions of enzymes in such a way that EC x is a more general description than EC x.y that in turn is a more general description than EC x.y.z. The top level EC numbers and names are listed below:

EC 1 oxidoreductases

EC 2 transferases

EC 3 hydrolases

EC 4 lyases

EC 5 isomerases

EC 6 ligases

The EC name for a given EC number can be viewed at http://www.chem.qmul.ac.uk/iupac/jcbn/index.html#6

Mappings between probe identifiers and enzyme identifiers were obtained using files provided by: KEGG GENOME ftp://ftp.genome.jp/pub/kegg/genomes With a date stamp from the source of: 2011-Mar15

References

ftp://ftp.genome.ad.jp/pub/kegg/pathways

Examples

```
x <- chickenENZYME
# Get the probe identifiers that are mapped to an EC number
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the ENZYME for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

```
# Now convert chickenENZYME2PROBE to a list to see inside
xx <- as.list(chickenENZYME2PROBE)
if(length(xx) > 0){
   # Get the probe identifiers for the first five enzyme
   #commission numbers
   xx[1:5]
   # Get the first one
   xx[[1]]
}
```

---

chickenGENENAME *Map between Manufacturer IDs and Genes*

Description

chickenGENENAME is an R object that maps manufacturer identifiers to the corresponding gene name.

Details

Each manufacturer identifier maps to a named vector containing the gene name. The vector name corresponds to the manufacturer identifier. If the gene name is unknown, the vector will contain an NA.

Gene names currently include both the official (validated by a nomenclature committee) and preferred names (interim selected for display) for genes. Efforts are being made to differentiate the two by adding a name to the vector.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

Examples

```
x <- chickenGENENAME
# Get the probe identifiers that are mapped to a gene name
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the GENENAME for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

chickenGO *Maps between manufacturer IDs and Gene Ontology (GO) IDs*

Description

chickenGO is an R object that provides mappings between manufacturer identifiers and the GO identifiers that they are directly associated with. This mapping and its reverse mapping (chickenGO2PROBE) do NOT associate the child terms from the GO ontology with the gene. Only the directly evidenced terms are represented here.

chickenGO2ALLPROBES is an R object that provides mappings between a given GO identifier and all of the manufacturer identifiers annotated at that GO term OR TO ONE OF IT'S CHILD NODES in the GO ontology. Thus, this mapping is much larger and more inclusive than chickenGO2PROBE.

Details

If chickenGO is cast as a list, each manufacturer identifier is mapped to a list of lists. The names on the outer list are GO identifiers. Each inner list consists of three named elements: GOID, Ontology, and Evidence.

The GOID element matches the GO identifier named in the outer list and is included for convenience when processing the data using 'lapply'.

The Ontology element indicates which of the three Gene Ontology categories this identifier belongs to. The categories are biological process (BP), cellular component (CC), and molecular function (MF).

The Evidence element contains a code indicating what kind of evidence supports the association of the GO identifier to the manufacturer id. The evidence codes in use include:

IMP: inferred from mutant phenotype

IGI: inferred from genetic interaction

IPI: inferred from physical interaction

ISS: inferred from sequence similarity

IDA: inferred from direct assay

IEP: inferred from expression pattern

IEA: inferred from electronic annotation

TAS: traceable author statement

NAS: non-traceable author statement

ND: no biological data available

IC: inferred by curator

If chickenGO2ALLPROBES or chickenGO2PROBE is cast as a list, each GO term maps to a named vector of manufacturer identifiers and evidence codes. A GO identifier may be mapped to the same manufacturer identifier more than once but the evidence code can be different. Mappings between Gene Ontology identifiers and Gene Ontology terms and other information are available in a separate data package named GO.

Whenever any of these mappings are cast as a data.frame, all the results will be output in an appropriate tabular form.

Mappings between manufacturer identifiers and GO information were obtained through their mappings to manufacturer identifiers. NAs are assigned to manufacturer identifiers that can not be mapped to any Gene Ontology information. Mappings between Gene Ontology identifiers an Gene Ontology terms and other information are available in a separate data package named GO.

All mappings were based on data provided by: Gene Ontology ftp://ftp.geneontology.org/pub/go/godatabase/archive/latest-lite/ With a date stamp from the source of: 20130302

References

ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/

See Also

chickenGO2ALLPROBES.

Examples

```
x <- chickenGO
# Get the manufacturer identifiers that are mapped to a GO ID
mapped_genes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_genes])
if(length(xx) > 0) {
    # Try the first one
    got <- xx[[1]]
    got[[1]][["GOID"]]
    got[[1]][["Ontology"]]
    got[[1]][["Evidence"]]
}
# For the reverse map:
# Convert to a list
xx <- as.list(chickenGO2PROBE)
if(length(xx) > 0){
    # Gets the manufacturer ids for the top 2nd and 3nd GO identifiers
    goids <- xx[2:3]
    # Gets the manufacturer ids for the first element of goids
    goids[[1]]
    # Evidence code for the mappings
    names(goids[[1]])
}
# Convert chickenGO2ALLPROBES to a list
xx <- as.list(chickenGO2ALLPROBES)
if(length(xx) > 0){
    # Gets the manufacturer identifiers for the top 2nd and 3nd GO identifiers
    goids <- xx[2:3]
    # Gets all the manufacturer identifiers for the first element of goids
    goids[[1]]
    # Evidence code for the mappings
    names(goids[[1]])
```

```
}
```

---

chickenMAPCOUNTS *Number of mapped keys for the maps in package chicken.db*

Description

chickenMAPCOUNTS provides the "map count" (i.e. the count of mapped keys) for each map in package chicken.db.

Details

This "map count" information is precalculated and stored in the package annotation DB. This allows some quality control and is used by the checkMAPCOUNTS function defined in AnnotationDbi to compare and validate different methods (like `count.mappedkeys(x)` or `sum(!is.na(as.list(x)))`) for getting the "map count" of a given map.

See Also

mappedkeys, count.mappedkeys, checkMAPCOUNTS

Examples

```
chickenMAPCOUNTS
mapnames <- names(chickenMAPCOUNTS)
chickenMAPCOUNTS[mapnames[1]]
x <- get(mapnames[1])
sum(!is.na(as.list(x)))
count.mappedkeys(x)   # much faster!

## Check the "map count" of all the maps in package chicken.db
checkMAPCOUNTS("chicken.db")
```

---

chickenORGANISM *The Organism information for chicken*

Description

chickenORGANISM is an R object that contains a single item: a character string that names the organism for which chicken was built. chickenORGPKG is an R object that contains a chararcter vector with the name of the organism package that a chip package depends on for its gene-centric annotation.

Details

Although the package name is suggestive of the organism for which it was built, chickenORGANISM provides a simple way to programmatically extract the organism name. chickenORGPKG provides a simple way to programmatically extract the name of the parent organism package. The parent organism package is a strict dependency for chip packages as this is where the gene cetric information is ultimately extracted from. The full package name will always be this string plus the extension ".db". But most programatic acces will not require this extension, so its more convenient to leave it out.

Examples

```
chickenORGANISM
chickenORGPKG
```

---

chickenPATH *Mappings between probe identifiers and KEGG pathway identifiers*

Description

KEGG (Kyoto Encyclopedia of Genes and Genomes) maintains pathway data for various organisms.

chickenPATH maps probe identifiers to the identifiers used by KEGG for pathways in which the genes represented by the probe identifiers are involved

chickenPATH2PROBE is an R object that provides mappings between KEGG identifiers and manufacturer identifiers.

Details

Each KEGG pathway has a name and identifier. Pathway name for a given pathway identifier can be obtained using the KEGG data package that can either be built using AnnBuilder or downloaded from Bioconductor http://www.bioconductor.org.

Graphic presentations of pathways are searchable at url http://www.genome.ad.jp/kegg/pathway.html by using pathway identifiers as keys.

Mappings were based on data provided by: KEGG GENOME ftp://ftp.genome.jp/pub/kegg/genomes With a date stamp from the source of: 2011-Mar15

References

http://www.genome.ad.jp/kegg/

Examples

```
x <- chickenPATH
# Get the probe identifiers that are mapped to a KEGG pathway ID
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the PATH for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}

# Now convert the chickenPATH2PROBE object to a list
xx <- as.list(chickenPATH2PROBE)
if(length(xx) > 0){
    # Get the probe identifiers for the first two pathway identifiers
    xx[1:2]
    # Get the first one
    xx[[1]]
}
```

---

chickenPFAM *Map Manufacturer IDs to Pfam IDs*

Description

chickenPFAM is an R object that provides mappings between a manufacturer identifier and the associated Pfam identifiers.

Details

Each manufacturer identifier maps to a named vector of Pfam identifiers. The name for each Pfam identifier is the IPI accession numbe where this Pfam identifier is found.

If the Pfam is a named NA, it means that the associated Entrez Gene id of this manufacturer identifier is found in an IPI entry of the IPI database, but there is no Pfam identifier in the entry.

If the Pfam is a non-named NA, it means that the associated Entrez Gene id of this manufacturer identifier is not found in any IPI entry of the IPI database.

Mappings were based on data provided by: Uniprot http://www.UniProt.org/ With a date stamp from the source of: Thu Mar 7 17:40:36 2013

Examples

```
x <- chickenPFAM
# Get the probe identifiers that are mapped to any Pfam ID
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
```

```
# randomly display 10 probes
sample(xx, 10)
```

---

chickenPMID *Maps between Manufacturer Identifiers and PubMed Identifiers*

Description

chickenPMID is an R object that provides mappings between manufacturer identifiers and PubMed identifiers. chickenPMID2PROBE is an R object that provides mappings between PubMed identifiers and manufacturer identifiers.

Details

When chickenPMID is viewed as a list each manufacturer identifier is mapped to a named vector of PubMed identifiers. The name associated with each vector corresponds to the manufacturer identifier. The length of the vector may be one or greater, depending on how many PubMed identifiers a given manufacturer identifier is mapped to. An NA is reported for any manufacturer identifier that cannot be mapped to a PubMed identifier.

When chickenPMID2PROBE is viewed as a list each PubMed identifier is mapped to a named vector of manufacturer identifiers. The name represents the PubMed identifier and the vector contains all manufacturer identifiers that are represented by that PubMed identifier. The length of the vector may be one or longer, depending on how many manufacturer identifiers are mapped to a given PubMed identifier.

Titles, abstracts, and possibly full texts of articles can be obtained from PubMed by providing a valid PubMed identifier. The pubmed function of annotate can also be used for the same purpose.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

References

http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed

Examples

```
x <- chickenPMID
# Get the probe identifiers that are mapped to any PubMed ID
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0){
# Get the PubMed identifiers for the first two probe identifiers
xx[1:2]
# Get the first one
xx[[1]]
if(interactive() && !is.null(xx[[1]]) && !is.na(xx[[1]])
   && require(annotate)){
    # Get article information as XML files
```

```
        xmls <- pubmed(xx[[1]], disp = "data")
        # View article information using a browser
        pubmed(xx[[1]], disp = "browser")
    }
}

# Now convert the reverse map object chickenPMID2PROBE to a list
xx <- as.list(chickenPMID2PROBE)
if(length(xx) > 0){
    # Get the probe identifiers for the first two PubMed identifiers
    xx[1:2]
    # Get the first one
    xx[[1]]
    if(interactive() && require(annotate)){
        # Get article information as XML files for a PubMed id
        xmls <- pubmed(names(xx)[1], disp = "data")
        # View article information using a browser
        pubmed(names(xx)[1], disp = "browser")
    }
}
```

---

chickenPROSITE *Map Manufacturer IDs to PROSITE ID*

Description

chickenPROSITE is an R object that provides mappings between a manufacturer identifier and the associated PROSITE identifiers.

Details

Each manufacturer identifier maps to a named vector of PROSITE identifiers. The name for each PROSITE identifier is the IPI accession numbe where this PROSITE identifier is found.

If the PROSITE is a named NA, it means that the associated Entrez Gene id of this manufacturer identifier is found in an IPI entry of the IPI database, but there is no PROSITE identifier in the entry.

If the PROSITE is a non-named NA, it means that the associated Entrez Gene id of this manufacturer identifier is not found in any IPI entry of the IPI database.

Mappings were based on data provided by: Uniprot http://www.UniProt.org/ With a date stamp from the source of: Thu Mar 7 17:40:36 2013

Examples

```
x <- chickenPROSITE
# Get the probe identifiers that are mapped to any PROSITE ID
mapped_probes <- mappedkeys(x)
# Convert to a list
xxx <- as.list(x[mapped_probes])
# randomly display 10 probes
xxx[sample(1:length(xxx), 10)]
```

chickenREFSEQ *Map between Manufacturer Identifiers and RefSeq Identifiers*

Description

chickenREFSEQ is an R object that provides mappings between manufacturer identifiers and RefSeq identifiers.

Details

Each manufacturer identifier is mapped to a named vector of RefSeq identifiers. The name represents the manufacturer identifier and the vector contains all RefSeq identifiers that can be mapped to that manufacturer identifier. The length of the vector may be one or greater, depending on how many RefSeq identifiers a given manufacturer identifier can be mapped to. An NA is reported for any manufacturer identifier that cannot be mapped to a RefSeq identifier at this time.

RefSeq identifiers differ in format according to the type of record the identifiers are for as shown below:

NG_XXXXX: RefSeq accessions for genomic region (nucleotide) records

NM_XXXXX: RefSeq accessions for mRNA records

NC_XXXXX: RefSeq accessions for chromosome records

NP_XXXXX: RefSeq accessions for protein records

XR_XXXXX: RefSeq accessions for model RNAs that are not associated with protein products

XM_XXXXX: RefSeq accessions for model mRNA records

XP_XXXXX: RefSeq accessions for model protein records

Where XXXXX is a sequence of integers.

NCBI http://www.ncbi.nlm.nih.gov/RefSeq/ allows users to query the RefSeq database using RefSeq identifiers.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

References

http://www.ncbi.nlm.nih.gov http://www.ncbi.nlm.nih.gov/RefSeq/

Examples

```
x <- chickenREFSEQ
# Get the probe identifiers that are mapped to any RefSeq ID
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the REFSEQ for the first five probes
  xx[1:5]
```

```
  # Get the first one
  xx[[1]]
}
```

---

chickenSYMBOL *Map between Manufacturer Identifiers and Gene Symbols*

Description

chickenSYMBOL is an R object that provides mappings between manufacturer identifiers and gene abbreviations.

Details

Each manufacturer identifier is mapped to an abbreviation for the corresponding gene. An NA is reported if there is no known abbreviation for a given gene.

Symbols typically consist of 3 letters that define either a single gene (ABC) or multiple genes (ABC1, ABC2, ABC3). Gene symbols can be used as key words to query public databases such as Entrez Gene.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

References

http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene

Examples

```
x <- chickenSYMBOL
# Get the probe identifiers that are mapped to a gene symbol
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the SYMBOL for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

chickenUNIGENE *Map between Manufacturer Identifiers and UniGene cluster identifiers*

Description

chickenUNIGENE is an R object that provides mappings between manufacturer identifiers and UniGene identifiers.

Details

Each manufacturer identifier is mapped to a UniGene identifier. An NA is reported if the manufacturer identifier cannot be mapped to UniGene at this time.

A UniGene identifier represents a cluster of sequences of a gene. Using UniGene identifiers one can query the UniGene database for information about the sequences or the Entrez Gene database for information about the genes.

Mappings were based on data provided by: Entrez Gene ftp://ftp.ncbi.nlm.nih.gov/gene/DATA With a date stamp from the source of: 2013-Mar5

References

http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene

Examples

```
x <- chickenUNIGENE
# Get the probe identifiers that are mapped to an UNIGENE ID
mapped_probes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_probes])
if(length(xx) > 0) {
  # Get the UNIGENE for the first five probes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

chickenUNIPROT *Map Uniprot accession numbers with Entrez Gene identifiers*

Description

chickenUNIPROT is an R object that contains mappings between the manufacturer identifiers and Uniprot accession numbers.

Details

This object is a simple mapping of manufacturer identifiers to Uniprot Accessions.

Mappings were based on data provided by NCBI (link above) with an exception for fly, which required retrieving the data from ensembl http://www.ensembl.org/biomart/martview/

Examples

```
x <- chickenUNIPROT
# Get the entrez gene IDs that are mapped to an Uniprot ID
mapped_genes <- mappedkeys(x)
# Convert to a list
xx <- as.list(x[mapped_genes])
if(length(xx) > 0) {
  # Get the Uniprot IDs for the first five genes
  xx[1:5]
  # Get the first one
  xx[[1]]
}
```

---

chicken_dbconn *Collect information about the package annotation DB*

Description

Some convenience functions for getting a connection object to (or collecting information about) the package annotation DB.

Usage

```
chicken_dbconn()
chicken_dbfile()
chicken_dbschema(file="", show.indices=FALSE)
chicken_dbInfo()
```

Arguments

file	A connection, or a character string naming the file to print to (see the file argument of the cat function for the details).
show.indices	The CREATE INDEX statements are not shown by default. Use show.indices=TRUE to get them.

Details

chicken_dbconn returns a connection object to the package annotation DB. IMPORTANT: Don't call dbDisconnect on the connection object returned by chicken_dbconn or you will break all the AnnDbObj objects defined in this package!

chicken_dbfile returns the path (character string) to the package annotation DB (this is an SQLite file).

chicken_dbschema prints the schema definition of the package annotation DB.

chicken_dbInfo prints other information about the package annotation DB.

Value

chicken_dbconn: a DBIConnection object representing an open connection to the package annotation DB.

chicken_dbfile: a character string with the path to the package annotation DB.

chicken_dbschema: none (invisible NULL).

chicken_dbInfo: none (invisible NULL).

See Also

dbGetQuery, dbConnect, dbconn, dbfile, dbschema, dbInfo

Examples

```
## Count the number of rows in the "probes" table:
dbGetQuery(chicken_dbconn(), "SELECT COUNT(*) FROM probes")

## The connection object returned by chicken_dbconn() was
## created with:
dbConnect(SQLite(), dbname=chicken_dbfile(), cache_size=64000,
synchronous=0)

chicken_dbschema()

chicken_dbInfo()
```

Index