Package ‘OrganismDbi’

October 9, 2015

Title Software to enable the smooth interfacing of different database packages

Description The package enables a simple unified interface to several annotation packages each of which has its own schema by taking advantage of the fact that each of these packages implements a select methods.

Version 1.10.0

Author Marc Carlson, Herve Pages, Martin Morgan, Valerie Obenchain

Maintainer Biocore Data Team <maintainer@bioconductor.org>

Depends R (>= 2.14.0), methods, AnnotationDbi (>= 1.16.10), GenomicFeatures (>= 1.17.13)

Imports BiocGenerics, graph, RBGL, AnnotationDbi, GenomicFeatures, stats

Suggests Homo.sapiens, Rattus.norvegicus, BSgenome.Hsapiens.UCSC.hg19, RUnit

Collate AllGenerics.R AllClasses.R methods-select.R methods-transcripts.R createOrganismPackage.R seqinfo.R test_OrganismDbi_package.R

License Artistic-2.0

biocViews Annotation, Infrastructure

NeedsCompilation no

R topics documented:

makeOrganismPackage 2
mapToTranscripts 3
OrganismDb-class 5
rangeBasedAccessors 7

Index **12**

makeOrganismPackage *Making OrganismDb packages from annotation packages.*

Description

makeOrganismPackage is a method that generates a package that will load an appropriate annotationOrganismDb object that will in turn point to existing annotation packages.

Usage

```
makeOrganismPackage (pkgname,
                     graphData,
                     organism,
                     version,
                     maintainer,
                     author,
                     destDir,
                     license="Artistic-2.0")
```

Arguments

pkgname	What is the desired package name. Traditionally, this should be the genus and species separated by a ".". So as an example, "Homo.sapiens" would be the package name for human
graphData	A list of short character vectors. Each character vector in the list is exactly two elements long and represents a join relationship between two packages. The names of these character vectors are the package names and the values are the foreign keys that should be used to connect each package. All foreign keys must be values that can be returned by the columns method for each package in question, and obviously they also must be the same kind of identifier as well.
organism	The name of the organism this package represents
version	What is the version number for this package?
maintainer	Who is the package maintainer? (must include email to be valid)
author	Who is the creator of this package?
destDir	A path where the package source should be assembled.
license	What is the license (and it's version)

Details

The purpose of this method is to create a special package that will depend on existing annotation packages and which will load a special annotationOrganismDb object that will allow proper dispatch of special select methods. These methods will allow the user to easily query across multiple annotation resources via information contained by the annotationOrganismDb object. Because the end result will be a package that treats all the data mapped together as a single source, the user is encouraged to take extra care to ensure that the different packages used are from the same build etc.

Value

A special package to load an OrganismDb object.

Author(s)

M. Carlson

See Also

`OrganismDb`

Examples

```
## set up the list with the relevant relationships:
gd <- list(join1 = c(GO.db="GOID", org.Hs.eg.db="GO"),
           join2 = c(org.Hs.eg.db="ENTREZID",
                     TxDb.Hsapiens.UCSC.hg19.knownGene="GENEID"))

## sets up a temporary directory for this example
## (users won't need to do this step)
destination <- tempfile()
dir.create(destination)

## makes an Organism package for human called Homo.sapiens
makeOrganismPackage(pkgname = "Homo.sapiens",
  graphData = gd,
  organism = "Homo sapiens",
  version = "1.0.0",
  maintainer = "Bioconductor Package Maintainer <maintainer@bioconductor.org>",
  author = "Bioconductor Core Team",
  destDir = destination,
  license = "Artistic-2.0")
```

`mapToTranscripts` *Map range coordinates between transcripts and genome space*

Description

Map range coordinates between features in the transcriptome and genome (reference) space.

See ?`mapToAlignments` in the **GenomicAlignments** package for mapping coordinates between reads (local) and genome (reference) space using a CIGAR alignment.

Usage

```
## S4 method for signature 'ANY,OrganismDb'
mapToTranscripts(x, transcripts,
          ignore.strand = TRUE,
          extractor.fun = GenomicFeatures::transcripts, ...)
```

Arguments

`x`	GRanges-class object of positions to be mapped. `x` must have names when mapping to the genome.
`transcripts`	The `OrganismDb` object that will be used to extract features using the `extractor.fun`.
`ignore.strand`	When TRUE, strand is ignored in overlap operations.
`extractor.fun`	Function to extract genomic features from a `TxDb` object. Valid `extractor` functions: • transcripts ## default • exons • cds • genes • promoters • disjointExons • microRNAs • tRNAs • transcriptsBy • exonsBy • cdsBy • intronsByTranscript • fiveUTRsByTranscript • threeUTRsByTranscript
`...`	Additional arguments passed to `extractor.fun` functions.

Details

- `mapToTranscripts` The genomic range in `x` is mapped to the local position in the `transcripts` ranges. A successful mapping occurs when `x` is completely within the `transcripts` range, equivalent to:

  ```
  findOverlaps(..., type="within")
  ```

  Transcriptome-based coordinates start counting at 1 at the beginning of the `transcripts` range and return positions where `x` was aligned. The seqlevels of the return object are taken from the `transcripts` object and should be transcript names. In this direction, mapping is attempted between all elements of `x` and all elements of `transcripts`.

Value

An object the same class as `x`.

Parallel methods return an object the same shape as `x`. Ranges that cannot be mapped (out of bounds or strand mismatch) are returned as zero-width ranges starting at 0 with a seqname of "UN-MAPPED".

Non-parallel methods return an object that varies in length similar to a Hits object. The result only contains mapped records, strand mismatch and out of bound ranges are not returned. `xHits` and `transcriptsHits` metadata columns indicate the elements of `x` and `transcripts` used in the mapping.

When present, names from `x` are propagated to the output. When mapping to transcript coordinates, seqlevels of the output are the names on the `transcripts` object; most often these will be transcript names. When mapping to the genome, seqlevels of the output are the seqlevels of `transcripts` which are usually chromosome names.

Author(s)

V. Obenchain, M. Lawrence and H. Pages; ported to work with OrganismDbi by Marc Carlson

See Also

- mapToTranscripts .

Examples

```
## ---------------------------------------------------------------------
## A. Basic Use
## ---------------------------------------------------------------------

library(Homo.sapiens)
x <- GRanges("chr5",
             IRanges(c(173315331,174151575), width=400,
                     names=LETTERS[1:2]))

## Map to transcript coordinates:
mapToTranscripts(x, Homo.sapiens)
```

`OrganismDb-class` *OrganismDb objects*

Description

The OrganismDb class is a container for storing knowledge about existing Annotation packages and the relationships between these resources. The purpose of this object and it's associated methods is to provide a means by which users can conveniently query for data from several different annotation resources at the same time using a familiar interface.

The supporting methods `select`, `columns` and `keys` are used together to extract data from an `OrganismDb` object in a manner that should be consistent with how these are used on the supporting annotation resources.

The family of `seqinfo` style getters (`seqinfo`, `seqlevels`, `seqlengths`, `isCircular`, `genome`, and `seqnameStyle`) is also supported for OrganismDb objects provided that the object in question has an embedded TxDb object.

Methods

In the code snippets below, `x` is a OrganismDb object. For the metadata and show methods, there is also support for FeatureDb objects.

`keytypes(x)`: allows the user to discover which keytypes can be passed in to `select` or `keys` and the `keytype` argument.

`keys(x, keytype, pattern, column, fuzzy)`: Return keys for the database contained in the TxDb object .

The `keytype` argument specifies the kind of keys that will be returned and is always required.

If `keys` is used with `pattern`, it will pattern match on the `keytype`.

But if the `column` argument is also provided along with the `pattern` argument, then `pattern` will be matched against the values in `column` instead.

If `keys` is called with `column` and no `pattern` argument, then it will return all keys that have corresponding values in the `column` argument.

Thus, the behavior of `keys` all depends on how many arguments are specified.

Use of the `fuzzy` argument will toggle fuzzy matching to TRUE or FALSE. If `pattern` is not used, fuzzy is ignored.

`columns(x)`: shows which kinds of data can be returned for the `OrganismDb` object.

`select(x, keys, columns, keytype)`: When all the appropriate arguments are specifiedm `select` will retrieve the matching data as a data.frame based on parameters for selected `keys` and `columns` and `keytype` arguments.

Author(s)

Marc Carlson

See Also

- AnnotationDb-class for more descriptsion of methods `select`,`keytypes`,`keys` and `columns`.
- makeOrganismPackage for functions used to generate an `OrganismDb` based package.
- rangeBasedAccessors for the range based methods used in extracting data from a `OrganismDb` object.
- seqlevels .
- seqlengths .
- isCircular .
- genome .

Examples

```
## load a package that creates an OrganismDb
library(Homo.sapiens)
ls(2)
## then the methods can be used on this object.
columns <- columns(Homo.sapiens)[c(7,10,11,12)]
keys <- head(keys(org.Hs.eg.db, "ENTREZID"))
keytype <- "ENTREZID"
```

```
res <- select(Homo.sapiens, keys, columns, keytype)
head(res)

## Get the DB connections or DB file paths associated with those for
## each.
dbconn(Homo.sapiens)
dbfile(Homo.sapiens)
```

---

rangeBasedAccessors *Extract genomic features from an object*

Description

Generic functions to extract genomic features from an object. This page documents the methods for OrganismDb objects only.

Usage

```
## S4 method for signature 'OrganismDb'
transcripts(x, vals=NULL, columns=c("TXID", "TXNAME"))

## S4 method for signature 'OrganismDb'
exons(x, vals=NULL, columns="EXONID")

## S4 method for signature 'OrganismDb'
cds(x, vals=NULL, columns="CDSID")

## S4 method for signature 'OrganismDb'
genes(x, vals=NULL, columns="GENEID")

## S4 method for signature 'OrganismDb'
transcriptsBy(x, by, columns, use.names=FALSE)

## S4 method for signature 'OrganismDb'
exonsBy(x, by, columns, use.names=FALSE)

## S4 method for signature 'OrganismDb'
cdsBy(x, by, columns, use.names=FALSE)

## S4 method for signature 'OrganismDb'
getTxDbIfAvailable(x, ...)



## S4 method for signature 'OrganismDb'
asBED(x)
## S4 method for signature 'OrganismDb'
```

```
asGFF(x)

## S4 method for signature 'OrganismDb'
disjointExons(x, aggregateGenes=FALSE,
              includeTranscripts=TRUE, ...)
## S4 method for signature 'OrganismDb'
microRNAs(x)
## S4 method for signature 'OrganismDb'
tRNAs(x)
## S4 method for signature 'OrganismDb'
promoters(x, upstream=2000, downstream=200, ...)

## S4 method for signature 'GenomicRanges,OrganismDb'
distance(x, y, ignore.strand=FALSE,
    ..., id, type=c("gene", "tx", "exon", "cds"))

## S4 method for signature 'BSgenome'
extractTranscriptSeqs(x, transcripts, strand = "+")

## S4 method for signature 'OrganismDb'
extractUpstreamSeqs(x, genes, width=1000, exclude.seqlevels=NULL)

## S4 method for signature 'OrganismDb'
intronsByTranscript(x, use.names=FALSE)
## S4 method for signature 'OrganismDb'
fiveUTRsByTranscript(x, use.names=FALSE)
## S4 method for signature 'OrganismDb'
threeUTRsByTranscript(x, use.names=FALSE)

## S4 method for signature 'OrganismDb'
isActiveSeq(x)
```

Arguments

`x`	A OrganismDb object. Except for the extractTranscriptSeqs method. In that case it's a BSgenome object and the second argument is an OrganismDb object.
`...`	Arguments to be passed to or from methods.
`by`	One of `"gene"`, `"exon"`, `"cds"` or `"tx"`. Determines the grouping.
`columns`	The columns or kinds of metadata that can be retrieved from the database. All possible columns are returned by using the `columns` method.
`use.names`	Controls how to set the names of the returned GRangesList object. These functions return all the features of a given type (e.g. all the exons) grouped by another feature type (e.g. grouped by transcript) in a GRangesList object. By default (i.e. if `use.names` is `FALSE`), the names of this GRangesList object (aka the group names) are the internal ids of the features used for grouping (aka the grouping features), which are guaranteed to be unique. If `use.names` is `TRUE`, then the names of the grouping features are used instead of their internal ids. For example, when grouping by transcript (`by="tx"`), the default group names

	are the transcript internal ids ("tx_id"). But, if use.names=TRUE, the group names are the transcript names ("tx_name"). Note that, unlike the feature ids, the feature names are not guaranteed to be unique or even defined (they could be all NAs). A warning is issued when this happens. See ?id2name for more information about feature internal ids and feature external names and how to map the formers to the latters. Finally, use.names=TRUE cannot be used when grouping by gene by="gene". This is because, unlike for the other features, the gene ids are external ids (e.g. Entrez Gene or Ensembl ids) so the db doesn't have a "gene_name" column for storing alternate gene names.
vals	Either NULL or a named list of vectors to be used to restrict the output. Valid names for this list are: "gene_id", "tx_id", "tx_name", "tx_chrom", "tx_strand", "exon_id", "exon_name", "exon_chrom", "exon_strand", "cds_id", "cds_name", "cds_chrom", "cds_strand" and "exon_rank".
upstream	For promoters : An integer(1) value indicating the number of bases upstream from the transcription start site. For additional details see ?`promoters,GRanges-method`.
downstream	For promoters : An integer(1) value indicating the number of bases downstream from the transcription start site. For additional details see ?`promoters,GRanges-method`.
aggregateGenes	For disjointExons : A logical. When FALSE (default) exon fragments that overlap multiple genes are dropped. When TRUE, all fragments are kept and the gene_id metadata column includes all gene ids that overlap the exon fragment.
includeTranscripts	For disjointExons : A logical. When TRUE (default) a tx_name metadata column is included that lists all transcript names that overlap the exon fragment.
y	For distance, a OrganismDb instance. The id is used to extract ranges from the OrganismDb which are then used to compute the distance from x.
id	A character vector the same length as x. The id must be identifiers in the OrganismDb object. type indicates what type of identifier id is.
type	A character(1) describing the id. Must be one of 'gene', 'tx', 'exon' or 'cds'.
ignore.strand	A logical indicating if the strand of the ranges should be ignored. When TRUE, strand is set to '+'.
transcripts	An object representing the exon ranges of each transcript to extract. It must be a GRangesList or OrganismDb object while the x is a BSgenome object. Internally, it's turned into a GRangesList object with exonsBy(transcripts, by="tx", use.names=TRUE).
strand	Only supported when x is a DNAString object. Can be an atomic vector, a factor, or an Rle object, in which case it indicates the strand of each transcript (i.e. all the exons in a transcript are considered to be on the same strand). More precisely: it's turned into a factor (or factor-Rle) that has the "standard strand levels" (this is done by calling the strand function on it). Then it's recycled to the length of RangesList object transcripts if needed. In the resulting object, the i-th element is interpreted as the strand of all the exons in the i-th transcript. strand can also be a list-like object, in which case it indicates the strand of each exon, individually. Thus it must have the same *shape* as RangesList object

	transcripts (i.e. same length plus strand[[i]] must have the same length as transcripts[[i]] for all i). strand can only contain "+" and/or "-" values. "*" is not allowed.
genes	An object containing the locations (i.e. chromosome name, start, end, and strand) of the genes or transcripts with respect to the reference genome. Only GenomicRanges and OrganismDb objects are supported at the moment. If the latter, the gene locations are obtained by calling the genes function on the OrganismDb object internally.
width	How many bases to extract upstream of each TSS (transcription start site).
exclude.seqlevels	A character vector containing the chromosome names (a.k.a. sequence levels) to exclude when the genes are obtained from a OrganismDb object.

Details

These are the range based functions for extracting transcript information from a OrganismDb object.

Value

a GRanges or GRangesList object

Author(s)

M. Carlson

See Also

- OrganismDb-class for how to use the simple "select" interface to extract information from a OrganismDb object.
- transcripts for the original transcripts method and related methods.
- transcriptsBy for the original transcriptsBy method and related methods.

Examples

```
## extracting all transcripts from Homo.sapiens with some extra metadata
library(Homo.sapiens)
cols = c("TXNAME","SYMBOL")
res <- transcripts(Homo.sapiens, columns=cols)

## extracting all transcripts from Homo.sapiens, grouped by gene and
## with extra metadata
res <- transcriptsBy(Homo.sapiens, by="gene", columns=cols)

## list possible values for columns argument:
columns(Homo.sapiens)

## Get the TxDb from an OrganismDb object (if it's available)
getTxDbIfAvailable(Homo.sapiens)
```

```
## Other functions listed above should work in way similar to their TxDb
## counterparts.  So for example:
promoters(Homo.sapiens)
## Should give the same value as:
promoters(getTxDbIfAvailable(Homo.sapiens))
```

Index

∗Topic **methods**
mapToTranscripts, 3
rangeBasedAccessors, 7
∗Topic **utilities**
mapToTranscripts, 3

AnnotationDb-class, *6*
asBED,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
asGFF,OrganismDb-method
*(*rangeBasedAccessors*)*, 7

BSgenome, *8, 9*

cds,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
cdsBy,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
class:OrganismDb *(*OrganismDb-class*)*, 5
columns,OrganismDb-method
*(*OrganismDb-class*)*, 5

dbconn,OrganismDb-method
*(*OrganismDb-class*)*, 5
dbfile,OrganismDb-method
*(*OrganismDb-class*)*, 5
disjointExons,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
distance,GenomicRanges,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
DNAString, *9*

exons,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
exonsBy, *9*
exonsBy,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
extractTranscriptSeqs,BSgenome-method
*(*rangeBasedAccessors*)*, 7
extractUpstreamSeqs,OrganismDb-method
*(*rangeBasedAccessors*)*, 7

fiveUTRsByTranscript,OrganismDb-method
*(*rangeBasedAccessors*)*, 7

genes, *10*
genes,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
genome, *6*
GenomicRanges, *10*
getTxDbIfAvailable
*(*rangeBasedAccessors*)*, 7
getTxDbIfAvailable,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
GRangesList, *8, 9*

id2name, *9*
intronsByTranscript,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
isActiveSeq,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
isActiveSeq<-,OrganismDb-method
*(*rangeBasedAccessors*)*, 7
isCircular, *6*

keys,OrganismDb-method
*(*OrganismDb-class*)*, 5
keytypes,OrganismDb-method
*(*OrganismDb-class*)*, 5

makeOrganismPackage, 2, *6*
mapToAlignments, *3*
mapToTranscripts, 3, *5*
mapToTranscripts,ANY,OrganismDb-method
*(*mapToTranscripts*)*, 3
metadata,OrganismDb-method
*(*OrganismDb-class*)*, 5
microRNAs,OrganismDb-method
*(*rangeBasedAccessors*)*, 7

OrganismDb, *3*, *7–10*
OrganismDb *(*OrganismDb-class*)*, 5
OrganismDb-class, 5, *10*

promoters,OrganismDb-method
(rangeBasedAccessors), 7

rangeBasedAccessors, *6*, 7
RangesList, *9*
Rle, *9*

select,OrganismDb-method
(OrganismDb-class), 5
seqinfo,OrganismDb-method
(OrganismDb-class), 5
seqlengths, *6*
seqlevels, *6*
strand, *9*

threeUTRsByTranscript,OrganismDb-method
(rangeBasedAccessors), 7
transcripts, *10*
transcripts,OrganismDb-method
(rangeBasedAccessors), 7
transcriptsBy, *10*
transcriptsBy,OrganismDb-method
(rangeBasedAccessors), 7
tRNAs,OrganismDb-method
(rangeBasedAccessors), 7
TxDb, *6*