Package ‘Organism.dplyr’

May 22, 2024

Title  dplyr-based Access to Bioconductor Annotation Resources

Version  1.32.0

Description  This package provides an alternative interface to
Bioconductor `annotation` resources, in particular the gene
identifier mapping functionality of the `org` packages (e.g.,
org.Hs.eg.db) and the genome coordinate functionality of the
`TxDb` packages (e.g., TxDb.Hsapiens.UCSC.hg38.knownGene).

Depends  R (>= 3.4), dplyr (>= 0.7.0), AnnotationFilter (>= 1.1.3)

Imports  RSQLite, S4Vectors, GenomeInfoDb, IRanges, GenomicRanges,
          GenomicFeatures, AnnotationDbi, rlang, methods, tools, utils,
          BiocFileCache, DBI, dbplyr, tibble

Suggests  org.Hs.eg.db, TxDb.Hsapiens.UCSC.hg38.knownGene,
           org.Mm.eg.db, TxDb.Mmusculus.UCSC.mm10.ensGene, testthat,
           knitr, rmarkdown, magick, BiocStyle, ggplot2

License  Artistic-2.0

Encoding  UTF-8

RoxygenNote  7.2.3


VignetteBuilder  knitr

biocViews  Annotation, Sequencing, GenomeAnnotation

BugReports  https://github.com/Bioconductor/Organism.dplyr/issues

git_url  https://git.bioconductor.org/packages/Organism.dplyr

git_branch  RELEASE_3_19

git_last_commit  0ac9982

git_last_commit_date  2024-04-30

Repository  Bioconductor 3.19

Date/Publication  2024-05-21

Author  Martin Morgan [aut, cre],
        Daniel van Twisk [ctb],
        Yubo Cheng [aut]
BasicFilter-class

Maintainer  Martin Morgan <martin.morgan@roswellpark.org>

Contents

BasicFilter-class ........................................... 2
Genomic-Extractors ........................................ 4
hg38light .................................................. 6
keytypes.src_organism-method .............................. 7
src_organism ............................................... 9

Index 12

BasicFilter-class  Filtering src_organism objects

Description

These functions create filters to be used by the "select" interface to src_organism objects.

Usage

AccnumFilter(value, condition = "==")
AliasFilter(value, condition = "==")
CdsChromFilter(value, condition = "==")
CdsIdFilter(value, condition = "==")
CdsNameFilter(value, condition = "==")
CdsStrandFilter(value, condition = "==")
EnsemblFilter(value, condition = "==")
EnsemblprotFilter(value, condition = "==")
EnsembltransFilter(value, condition = "==")
EnzymeFilter(value, condition = "==")
EvidenceFilter(value, condition = "==")
EvidenceallFilter(value, condition = "==")
ExonChromFilter(value, condition = "==")
ExonStrandFilter(value, condition = "==")
FlybaseFilter(value, condition = "==")
FlybaseCgFilter(value, condition = "==")
FlybaseProtFilter(value, condition = "==")
GeneChromFilter(value, condition = "==")
GeneStrandFilter(value, condition = "==")
GoFilter(value, condition = "==")
GoallFilter(value, condition = "==")
IpiFilter(value, condition = "==")
MapFilter(value, condition = "==")
MgiFilter(value, condition = "==")
OmimFilter(value, condition = "==")
OntologyFilter(value, condition = "==")
BasicFilter-class

```
OntologyallFilter(value, condition = "==")
PfamFilter(value, condition = "==")
PmidFilter(value, condition = "==")
PrositeFilter(value, condition = "==")
RefseqFilter(value, condition = "==")
TxChromFilter(value, condition = "==")
TxStrandFilter(value, condition = "==")
TxTypeFilter(value, condition = "==")
WormbaseFilter(value, condition = "==")
ZfinFilter(value, condition = "==")
```

## S4 method for signature 'BasicFilter'
show(object)

## S4 method for signature 'src_organism'
supportedFilters(object)

### Arguments

- `object` A BasicFilter or GRangesFilter object
- `value` Value of the filter. For GRangesFilter value should be a GRanges object.
- `condition` The condition to be used in filter for genomic extractors, one of "==", "!=",
  "startsWith" and "endsWith" are allowed, for numeric values (CdsStartFilter,
  CdsEndFilter, ExonStartFilter, ExonEndFilter, GeneStartFilter, GeneEndFilter,
  TxStartFilter and TxEndFilter), ",==", ",!=",
  ",>", ",>=", ",<" and ",<=". Default condition is "==".

### Details

All filters except GRangesFilter() takes value(s) from corresponding fields in the data base. For example, AccnumFilter() takes values of accession number(s), which come from field accnum. See keytypes() and keys() for possible values.

GRangesFilter() takes a GRanges object as filter, and returns genomic extractors (genes, transcripts, etc.) that are partially overlapping with the region.

supportedFilters() lists all available filters for src_organism object.

### Value

A Filter object showing class, value and condition of the filter

### Author(s)

Yubo Cheng.
Genomic-Extractors

See Also

src_organism for creating a src_organism object.
transcripts_tbl for generic functions to extract genomic features from a src_organism object.
select,src_organism-method for "select" interface on src_organism objects.

Examples

src <- src_organism(dbpath=hg38light())
keytypes(src)
head(keys(src, "ensembl"))

## filter by ensembl
EnsemblFilter("ENSG00000171862")

## filter by gene symbol start with "BRAC"
SymbolFilter("BRCA", "startsWith")

## filter by GRanges
GRangesFilter(GenomicRanges::GRanges("chr10:87869000-87876000"))

## filter by transcript start position
TxStartFilter(87863438, ">")

Genomic-Extractors Extract genomic features from src_organism objects

Description

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

These are the main functions for extracting transcript information from a src_organism object, inherited from transcripts in GenomicFeatures package. Two versions of results are provided: tibble (transcripts_tbl()) and GRanges or GRangesList (transcripts()).

Usage

cds(x, ...)
extons(x, ...)
genex(x, ...)
transcripts(x, ...)
cds_tbl(x, filter=NULL, columns=NULL)
extons_tbl(x, filter=NULL, columns=NULL)
genex_tbl(x, filter=NULL, columns=NULL)
transcripts_tbl(x, filter=NULL, columns=NULL)
cdsBy(x, by=c("tx", "gene"), ...)
extonsBy(x, by=c("tx", "gene"), ...)
transcriptsBy(x, by=c("gene", "exon", "cds"), ...)  
cdsBy_tbl(x, by=c("tx", "gene"), filter=NULL, columns=NULL)  
exonsBy_tbl(x, by=c("tx", "gene"), filter=NULL, columns=NULL)  
transcriptsBy_tbl(x, by=c("gene", "exon", "cds"), filter=NULL, columns=NULL)  
promoters_tbl(x, upstream, downstream, filter=NULL, columns=NULL)  
intronsByTranscript_tbl(x, filter=NULL, columns=NULL)  
fiveUTRsByTranscript(x, ...)  
fiveUTRsByTranscript_tbl(x, filter=NULL, columns=NULL)  
threeUTRsByTranscript(x, ...)  
threeUTRsByTranscript_tbl(x, filter=NULL, columns=NULL)

## S4 method for signature 'src_organism'  
promoters(x, upstream, downstream, filter = NULL, columns = NULL)

## S4 method for signature 'src_organism'  
intronsByTranscript(x, filter = NULL, columns = NULL)

Arguments

- **x**: A src_organism object
- **upstream**: For promoters(): An integer(1) value indicating the number of bases upstream from the transcription start site.
- **downstream**: For promoters(): An integer(1) value indicating the number of bases downstream from the transcription start site.
- **filter**: Either NULL, AnnotationFilter, or AnnotationFilterList to be used to restrict the output. Filters consists of AnnotationFilters and can be a GRanges object using "GRangesFilter" (see examples).
- **columns**: A character vector indicating columns to be included in output GRanges object or tbl.
- **by**: One of "gene", "exon", "cds" or "tx". Determines the grouping.
- **...**: Additional arguments to S4methods. In this case, the same as filter.

Value

functions with _tbl return a tibble object, other methods return a GRanges or GRangesList object.

Author(s)

Yubo Cheng.

See Also

src_organism for creating a src_organism object.
Examples

```r
## Not run: src <- src_ucsc("human")
src <- src_organism(dbpath=hg38light())

## transcript coordinates with filter in tibble format
filters <- AnnotationFilter(~symbol == c("A1BG", "CDH2"))
transcripts_tbl(src, filters)

transcripts_tbl(src, AnnotationFilter(~symbol %startsWith% "SNORD"))
transcripts_tbl(src, AnnotationFilter(~go == "GO:0005615"))
transcripts_tbl(src, filter=AnnotationFilter(~symbol %startsWith% "SNORD" & tx_start < 25070000))

## transcript coordinates with filter in granges format
filters <- GRangesFilter(GenomicRanges::GRanges("chr15:1-25070000"))
transcripts(src, filters)

## promoters
promoters(src, upstream=100, downstream=50,
          filter = SymbolFilter("ADA"))

## transcriptsBy
transcriptsBy(src, by = "exon", filter = SymbolFilter("ADA"))

## exonsBy
exonsBy(src, filter = SymbolFilter("ADA"))

## intronsByTranscript
intronsByTranscript(src, filter = SymbolFilter("ADA"))
```

---

**hg38light**

**Utilities used in examples, vignettes, and tests**

Description

These functions are primarily for illustrating functionality. `hg38light()` and `mm10light()` provide access to trimmed-down versions of Organism.dplyr data based derived from the TxDb.Hsapiens.UCSC.hg38.knownGene and TxDb.Mmusculus.UCSC.mm10.ensGene data bases.

Usage

```
hg38light()

mm10light()
```

Value

character(1) file path to the trimmed-down data base
Examples

hg38light()
mm10light()

Using the "select" interface on src_organism objects

Description

select, columns and keys can be used together to extract data from a src_organism object.

Usage

```r
## S4 method for signature 'src_organism'
keytypes(x)

## S4 method for signature 'src_organism'
columns(x)

## S4 method for signature 'src_organism'
keys(x, keytype, ...)

select_tbl(x, keys, columns, keytype)

## S4 method for signature 'src_organism'
select(x, keys, columns, keytype)

## S4 method for signature 'src_organism'
mapIds(x, keys, column, keytype, ..., multiVals)
```

Arguments

- **x**: a src_organism object
- **keytype**: specifies the kind of keys that will be returned. By default keys will return the keys for schema of the src_organism object.
- **...**: other arguments. These include:
  - pattern: the pattern to match.
  - column: the column to search on.
  - fuzzy: TRUE or FALSE value. Use fuzzy matching? (this is used with pattern)
- **keys**: the keys to select records for from the database. All possible keys are returned by using the keys method.
- **columns**: the columns or kinds of things that can be retrieved from the database. As with keys, all possible columns are returned by using the columns method.
column character(1) the column to search on, can only have a single element for the value

multiVals What should mapIds do when there are multiple values that could be returned. Options include:
first: when there are multiple matches only the 1st thing that comes back will be returned. This is the default behavior.
list: return a list object to the end user
filter: remove all elements that contain multiple matches and will therefore return a shorter vector than what came in whenever some of the keys match more than one value
asNA: return an NA value whenever there are multiple matches
CharacterList: returns a SimpleCharacterList object
FUN: can also supply a function to the multiVals argument for custom behaviors. The function must take a single argument and return a single value. This function will be applied to all the elements and will serve a 'rule' that for which thing to keep when there is more than one element. So for example this example function will always grab the last element in each result: \texttt{last <- function(x){x[length(x)]}}

Details
keytypes(): discover which keytypes can be passed to keytype argument of methods select or keys.
keys(): returns keys for the src_organism object. By default it returns the primary keys for the database, and returns the keys from that keytype when the keytype argument is used.
columns(): discover which kinds of data can be returned for the src_organism object.
select(): retrieves the data as a tibble based on parameters for selected keys columns and keytype arguments. If requested columns that have multiple matches for the keys, ‘select()’ will return a tibble with one row for each possible match.
mapIds(): gets the mapped ids (column) for a set of keys that are of a particular keytype. Usually returned as a named character vector.

Value
keys, columns and keytypes each returns a character vector of possible values. select returns a tibble.

Author(s)
Yubo Cheng.

See Also
AnnotationDb-class for more description of methods select, keytypes, keys and columns.
src_organism for creating a src_organism object.
transcripts_tbl for generic functions to extract genomic features from a src_organism object.
src_organism

Examples

```r
## Not run: src <- src_organism("TxDb.Hsapiens.UCSC.hg38.knownGene")
src <- src_organism(dbpath=hg38light())

## keytypes
keytypes(src)

## columns
columns(src)

## keys
keys(src, "entrez")

keytype <- "symbol"
keys <- c("ADA", "NAT2")
columns <- c("entrez", "tx_id", "tx_name", "exon_id")

## select
select_tbl(src, keys, columns, keytype)
select(src, keys, columns, keytype)

## mapIds
mapIds(src, keys, column = "tx_name", keytype)
```

Description

The database provides a convenient way to map between gene, transcript, and protein identifiers.

‘select_.tbl_organism()’ is DEPRECATED, please use ‘select()’.

Usage

```r
src_organism(txdb = NULL, dbpath = NULL, overwrite = FALSE)
src_ucsc(organism, genome = NULL, id = NULL, dbpath = NULL, verbose = TRUE)
supportedOrganisms()

## S3 method for class 'tbl_organism'
select_(.data, ...)

## S3 method for class 'src_organism'
src_tbls(x, ...)

## S3 method for class 'src_organism'
```
## Arguments

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>txdb</code></td>
<td>character(1) naming a TxDb.* package (e.g., <code>TxDb.Hsapiens.UCSC.hg38.knownGene</code>) or a TxDb object instantiating the content of a TxDb.* package.</td>
</tr>
<tr>
<td><code>dbpath</code></td>
<td>character(1) path or BiocFileCache instance representing the location where an Organism.dplyr SQLite database will be accessed or created. If no path is specified, the SQLite file is created in the default BiocFileCache() location.</td>
</tr>
<tr>
<td><code>overwrite</code></td>
<td>logical(1) overwrite an existing <code>dbpath</code> contains an Organism.dplyr SQLite database different from the version implied by <code>txdb</code>?</td>
</tr>
<tr>
<td><code>organism</code></td>
<td>organism or common name</td>
</tr>
<tr>
<td><code>genome</code></td>
<td>genome name</td>
</tr>
<tr>
<td><code>id</code></td>
<td>choose from &quot;knownGene&quot;, &quot;ensGene&quot; and &quot;refGene&quot;</td>
</tr>
<tr>
<td><code>verbose</code></td>
<td>logical. Should R report extra information on progress? Default is TRUE.</td>
</tr>
<tr>
<td><code>.data</code></td>
<td>A tbl.</td>
</tr>
<tr>
<td><code>x</code></td>
<td>A src_organism object</td>
</tr>
<tr>
<td><code>src</code></td>
<td>An src_organism object</td>
</tr>
</tbody>
</table>
| `from`    | character(1) name of temporary table in `src`.

## Details

`src_organism()` and `src_ucsc()` are meant to be a building block for `src_organism`, which provides an integrated presentation of identifiers and genomic coordinates.

`src_organism()` creates a dplyr database integrating org.* and TxDb.* information by given TxDb. And `src_ucsc()` creates the database by given organism name, genome and/or id.

`supportedOrganisms()` provides all supported organisms in this package with corresponding OrgDb and TxDb.

The `tbl(src_organism)` parameter `load_tbl_only` has been removed. The function behaves as `load_tbl_only = FALSE` (the previous default); for `load_tbl_only = TRUE`, use `tbl(src$con, ...)`.

## Value

`src_organism()` and `src_ucsc()` returns a dplyr `src_dbi` instance representing the data tables. A tibble of the requested table coming from the temporary database of the `src_organism` object.
Author(s)

Yubo Cheng.

See Also

dplyr for details about using dplyr to manipulate data.
transcripts_tbl for generic functions to extract genomic features from a src_organism object.
select,src_organism-method for "select" interface on src_organism objects.

Examples

```r
## create human sqlite database with TxDb.Hsapiens.UCSC.hg38.knownGene and
## corresponding org.Hs.eg.db
## Not run: src <- src_organism("TxDb.Hsapiens.UCSC.hg38.knownGene")
src <- src_organism(dbpath=hg38light())

## query using dplyr
inner_join(tbl(src, "id"), tbl(src, "id_go")) %>%
  filter(symbol == "ADA") %>%
dplyr::select(entrez, ensembl, symbol, go, evidence, ontology)

## create human sqlite database using hg38 genome
## Not run: human <- src_ucsc("human")

## all supported organisms with corresponding OrgDb and TxDb
supportedOrganisms()

## Look at all available tables
src_tbs(src)

## Look at data in table "id"
tbl(src, "id")

## Look at fields of one table
colnames(tbl(src, "id"))

## name of org package of src_organism object
orgPackageName(src)

## seqinfo of src_organism object
seqinfo(src)
```
Index

<table>
<thead>
<tr>
<th>Item</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccnumFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>AliasFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>BasicFilter-class</td>
<td>2</td>
</tr>
<tr>
<td>cds (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>cds_tbl (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>cdsBy (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>cdsBy_tbl (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>CdsChromFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>CdsIdFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>CdsNameFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>CdsStrandFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>CharacterFilter-class (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>columns,src_organism-method (keytypes,src_organism-method)</td>
<td>7</td>
</tr>
<tr>
<td>dplyr</td>
<td>11</td>
</tr>
<tr>
<td>EnsemblFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>EnsemblprotFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>EnsembltransFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>EnzymeFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>EvidenceallFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>EvidenceFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>ExonChromFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>exons (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>exons_tbl (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>exonsBy (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>exonsBy_tbl (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>ExonStrandFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>fiveUTRsByTranscript (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>fiveUTRsByTranscript_tbl (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>FlybaseCgFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>FlybaseFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>FlybaseProtFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>GeneChromFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>genes (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>genes_tbl (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>GeneStrandFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>Genomic-Extractors</td>
<td>4</td>
</tr>
<tr>
<td>GoallFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>GoFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>GRanges</td>
<td>4, 5</td>
</tr>
<tr>
<td>GRangesList</td>
<td>4, 5</td>
</tr>
<tr>
<td>hg38light</td>
<td>6</td>
</tr>
<tr>
<td>IntegerFilter-class (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>intronsByTranscript (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>intronsByTranscript,src_organism-method (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>intronsByTranscript_tbl (Genomic-Extractors)</td>
<td>4</td>
</tr>
<tr>
<td>IpiFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>keys,src_organism-method (keytypes,src_organism-method)</td>
<td>7</td>
</tr>
<tr>
<td>keytypes,src_organism-method</td>
<td>7</td>
</tr>
<tr>
<td>MapFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>mapIds,src_organism-method (keytypes,src_organism-method)</td>
<td>7</td>
</tr>
<tr>
<td>MgiFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>mm10light (hg38light)</td>
<td>6</td>
</tr>
<tr>
<td>OmimFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>OntologyallFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>OntologyFilter (BasicFilter-class)</td>
<td>2</td>
</tr>
<tr>
<td>orgPackageName,src_organism-method (src_organism)</td>
<td>9</td>
</tr>
</tbody>
</table>
INDEX

PfamFilter (BasicFilter-class), 2
PmidFilter (BasicFilter-class), 2
promoters (Genomic-Extractors), 4
promoters.src_organism-method (Genomic-Extractors), 4
promoters_tbl (Genomic-Extractors), 4
PrositeFilter (BasicFilter-class), 2
RefseqFilter (BasicFilter-class), 2
select.src_organism-method (keytypes, src_organism-method), 7
select._tbl_organism (src_organism), 9
select_tbl (keytypes, src_organism-method), 7
seqinfo.src_organism-method (src_organism), 9
show.BasicFilter-method (BasicFilter-class), 2
show.CharacterFilter-method (BasicFilter-class), 2
show_IntegerFilter-method (BasicFilter-class), 2
src_organism, 4, 5, 7, 8, 9, 10
src_tbls.src_organism (src_organism), 9
src_ucsc (src_organism), 9
supportedFilters.src_organism-method (BasicFilter-class), 2
supportedOrganisms (src_organism), 9
tbl.src_organism (src_organism), 9
threeUTRsByTranscript (Genomic-Extractors), 4
threeUTRsByTranscript_tbl (Genomic-Extractors), 4
tibble, 4, 5
transcripts, 4
transcripts (Genomic-Extractors), 4
transcripts_tbl, 4, 8, 11
transcripts_tbl (Genomic-Extractors), 4
transcriptsBy (Genomic-Extractors), 4
transcriptsBy_tbl (Genomic-Extractors), 4
TxChromFilter (BasicFilter-class), 2
TxStrandFilter (BasicFilter-class), 2
TxTypeFilter (BasicFilter-class), 2
WormbaseFilter (BasicFilter-class), 2