Package 'paxtoolsr'

March 13, 2017

Type Package

Title PaxtoolsR: Access Pathways from Multiple Databases through BioPAX and Pathway Commons

Version 1.8.0

Date 2016-05-13

Author Augustin Luna

Maintainer Augustin Luna <lunaa@cbio.mskcc.org>

Imports httr, igraph, plyr, rjson, R.utils, data.table

Depends R (>= 3.2), rJava (>= 0.9-4), XML

Suggests testthat, knitr, BiocStyle, rmarkdown, RColorBrewer, biomaRt, estrogen, affy, hgu95av2, hgu95av2cdf, limma, foreach, doSNOW, parallel

SystemRequirements Java (>= 1.6)

License LGPL-3

Description The package provides a set of R functions for interacting with BioPAX OWL files using Paxtools and the querying Pathway Commons (PC) molecular interaction database that are hosted by the Computational Biology Center at Memorial Sloan-Kettering Cancer Center (MSKCC). Pathway Commons databases include: BIND, BioGRID, CORUM, CTD, DIP, DrugBank, HPRD, HumanCyc, IntAct, KEGG, MirTarBase, Panther, PhosphoSitePlus, Reactome, RECON, TRANSFAC.

VignetteBuilder knitr

LazyData true

biocViews GeneSetEnrichment, GraphAndNetwork, Pathways, Software, SystemsBiology, NetworkEnrichment, Network, Reactome, KEGG

URL https://github.com/BioPAX/paxtoolsr

RoxygenNote 5.0.1

NeedsCompilation no

R topics documented:

addAttributeList *Add attributes using a list of vectors to an igraph object*

Description

Add attributes using a list of vectors to an igraph object

Usage

```
addAttributeList(g, attr, l)
```

Arguments

g	an igraph object
attr	the name of the attribute
l	the list of vectors

Value

the modified igraph object

convertSifnxIds *Convert IDs in a SIFNX*

Description

Convert IDs in a SIFNX

Usage

```
convertSifnxIds(sifnx, participantType = "ProteinReference",
  idType = "NCBI Gene", mapping = NULL, naRm = TRUE)
```

Arguments

sifnx	a SIFNX object (e.g. from the downloadPc2 function)
participantType	the type of participant on which the conversion will occur. Important because not all ID types apply to all entities and otherwise those entities would be labeled as missing an ID.
idType	an ID type for conversion (not used if mapping parameter used)
mapping	a two column data.frame with columns mapping$PARTICIPANT (old IDs to convert from) and mapping$ID (new IDs to convert to)
naRm	remove edges where NA's were introduced due to failed conversions

Value

a SIFNX list with nodes and edges. Only edges will have converted IDs

`convertSifToSpia` *Convert SIF Interaction Types to SPIA types*

Description

Convert SIF Interaction Types to SPIA types

Usage

```
convertSifToSpia(edges)
```

Arguments

`edges`	a data.frame of interactions; must have INTERACTION_TYPE column

Value

the edges data.frame with the converted interaction types

`convertToDF` *Convert Results from readSifnx to data.frame*

Description

Convert Results from readSifnx to data.frame

Usage

```
convertToDF(lst)
```

Arguments

`lst`	a list returned from readSifnx

Value

a list entries converted to data.frame

convertToDT *Convert Results from readSifnx to data.table*

Description

Convert Results from readSifnx to data.table

Usage

```
convertToDT(lst)
```

Arguments

lst	a list returned from readSifnx

Details

The SIFNX format is an evolving format. Older datasets may not have all the columns this function expects. In these cases, the columns will be added with all NULL entries.

Value

a list entries converted to data.table

downloadFile *Check Cache and Download File*

Description

Check Cache and Download File

Usage

```
downloadFile(baseUrl, fileName, destDir = NULL,
  cacheEnv = "PAXTOOLSR_CACHE", verbose = FALSE)
```

Arguments

baseUrl	a string, entire download URL except filename
fileName	a string, the filename of file to be downloaded
destDir	a string, the path where a file should be saved
cacheEnv	a string, the environment variable that points to the specific cache
verbose	show debugging information

Details

Description of file formats: http://www.pathwaycommons.org/pc2/formats

Value

a boolean TRUE if the file was downloaded or already exists, FALSE otherwise

See Also

readSif, readBiopax, readSbgn, readSifnx, readGmt

Examples

```
downloadFile("http://google.com/", fileName="index.html", destDir=tempdir())
```

---

downloadPc2 *Download Pathway Commons files (uses menu and cache)*

Description

Download Pathway Commons files (uses menu and cache)

Usage

```
downloadPc2(selectedFileName = NULL, destDir = NULL, returnNames = NULL,
  version = "current", verbose = FALSE)
```

Arguments

selectedFileName	a string, a name of a file to skip the the interactive selection
destDir	a string, the destination directory for the file to be downloaded (Default: NULL). If NULL, then file will be downloaded to cache directory file.path(Sys.getenv("HOME"), ".paxtoolsRCache")
returnNames	return a vector of names matching the given regular expression
version	a version number for a previous version of Pathway Commons data; versions 3 and above
verbose	a flag to display debugging information (Default: FALSE)

Value

an R object using one of the read* methods provided in this package corresponding to the file downloaded

Examples

```
## Not run:
  downloadPc2()
  downloadPc2(returnNames="ext.*sif")
  downloadPc2("PathwayCommons.8.inoh.GSEA.hgnc.gmt.gz", verbose=TRUE)

## End(Not run)
```

downloadSignedPC *Download a SIF file containing only signed interactions*

Description

Download a SIF file containing only signed interactions

Usage

```
downloadSignedPC(destDir = NULL)
```

Arguments

destDir	a string, the destination directory for the file to be downloaded (Default: NULL). If NULL, then file will be downloaded to cache directory file.path(Sys.getenv("HOME"), ".paxtoolsRCache")

Value

a SIF containing interactions that are considered signed (i.e. interactions causing an increase on decrease in a molecular species)

Examples

```
# downloadSignedPC()
```

extractIds *Extract IDs from an Extended SIF*

Description

Extract IDs from an Extended SIF

Usage

```
extractIds(nodes, participantType = "ProteinReference",
  idType = "hgnc symbol")
```

Arguments

nodes	extended SIF nodes entries as a data.table (from convertToDT)
participantType	a vector of types of participants to search; useful to only search protein (ProteinReference) or small molecule (SmallMoleculeReference) related entries.
idType	the type of ID to search for; case-insensitive

Details

IMPORTANT: Only the first matching ID will be returned. In some cases, multiple IDs will exist.

Value

a named vector of the first matches for the given ID type

Examples

```
tmp <- readSifnx(system.file("extdata", "test_sifnx.txt", package="paxtoolsr"))
results <- extractIds(tmp$nodes)
```

`fetch` *Fetch a set of IDs from a BioPAX OWL file*

Description

This function will create a subsetted object with specified URIs.

Usage

```
fetch(inputFile, outputFile = NULL, idList)
```

Arguments

`inputFile`	a string of the name of the input BioPAX OWL file
`outputFile`	a string with the name of the output BioPAX OWL file
`idList`	a vector of IDs from the BioPAX OWL file

Details

Only entities in the input BioPAX file will be used in the fetch. IDs used must be URIs for the entities of interest. Additional properties such as cross-references for fetched entities will be included in the output.

Value

an XMLInternalDocument representing a BioPAX OWL file

Examples

```
outFile <- tempfile()
ids <- c("http://identifiers.org/uniprot/P36894",
         "http://identifiers.org/uniprot/Q13873")
results <- fetch(system.file("extdata", "REACT_12034-3.owl", package="paxtoolsr"),
                 outFile, ids)
```

filterSif *Keep interactions in SIF network based on certain criteria*

Description

Keep interactions in SIF network based on certain criteria

Usage

```
filterSif(sif, interactionTypes = NULL, dataSources = NULL, ids = NULL,
  edgelist = NULL)
```

Arguments

sif	a binary SIF as a data.frame with three columns: "PARTICIPANT_A", "INTERACTION_TYPE", "PARTICIPANT_B"
interactionTypes	a vector of interaction types to be kept (List of interaction types: http://www.pathwaycommons.org/pc
dataSources	a vector of data sources to be kept
ids	a vector of IDs to be kept
edgelist	a two-column data.frame where each row is an interaction to be kept. Directionality is ignored (e.g. Edge A B will return interactions A B and B A from SIF)

Value

filtered interactions with three columns: "PARTICIPANT_A", "INTERACTION_TYPE", "PARTICIPANT_B". The intersection of multiple filters is returned. The return class is the same as the input: data.frame or data.table

Examples

```
results <- readSif(system.file("extdata", "test_sif.txt", package="paxtoolsr"))
intTypes <- c("controls-state-change-of", "controls-expression-of", "catalysis-precedes")
filteredNetwork <- filterSif(results, intTypes)

tmp <- readSifnx(system.file("extdata", "test_sifnx_250.txt", package = "paxtoolsr"))
results <- filterSif(tmp$edges, dataSources=c("INOH", "KEGG"))
results <- filterSif(tmp$edges, ids=c("CHEBI:17640", "MCM3"))
results <- filterSif(tmp$edges, dataSources=c("IntAct"), ids=c("CHEBI:17640", "MCM3"))

tmp <- readSifnx(system.file("extdata", "test_sifnx_250.txt", package = "paxtoolsr"))
edgelist <- read.table(system.file("extdata", "test_edgelist.txt", package = "paxtoolsr"),
  sep="\t", header=FALSE, stringsAsFactors=FALSE)
results <- filterSif(tmp$edges, edgelist=edgelist)
```

getCacheFiles *List files in cache directory*

Description

List files in cache directory

Usage

```
getCacheFiles()
```

Value

a vector of the files in the cache directory

Examples

```
getCacheFiles()
```

getErrorMessage *Get Error Message for a Pathway Commons Error*

Description

Get Error Message for a Pathway Commons Error

Usage

```
getErrorMessage(code)
```

Arguments

code	a three digit numerical error code

Value

an error message for the code

Examples

```
results <- getErrorMessage("452")
```

getNeighbors *Get the neighbors of a set of IDs in a BioPAX file*

Description

This function retrieves a set of neighbors for a set of IDs in a BioPAX file.

Usage

```
getNeighbors(inputFile, outputFile = NULL, idList)
```

Arguments

inputFile	a string with the name of the input BioPAX OWL file
outputFile	a string with the name of the output BioPAX OWL file
idList	a vector of IDs from the BioPAX OWL file

Details

Only entities in the input BioPAX file will be searched for neighbors. IDs used must be URIs for the entities of interest.

Value

an XMLInternalDocument representing a BioPAX OWL file

Examples

```
outFile <- tempfile()
results <- getNeighbors(system.file("extdata",
  "raf_map_kinase_cascade_reactome.owl", package="paxtoolsr"),
  outFile,
  c("HTTP://WWW.REACTOME.ORG/BIOPAX/48887#PROTEIN2360_1_9606",
    "HTTP://WWW.REACTOME.ORG/BIOPAX/48887#PROTEIN1631_1_9606"))
```

getPc *Get Pathway Commons BioPAX elements*

Description

This command retrieves full pathway information for a set of elements such as pathway, interaction or physical entity given the RDF IDs.

Usage

```
getPc(uri, format = "BIOPAX", verbose = FALSE)
```

Arguments

uri	a vector that includes valid/existing BioPAX element's URI (RDF ID; for utility classes that were "normalized", such as entity refereneces and controlled vocabularies, it is usually a Idntifiers.org URL. Multiple IDs are allowed per query, for example, c("http://identifiers.org/uniprot/Q06609", "http://identifiers.org/uniprot/Q549Z0") See also about MIRIAM and Identifiers.org in details.
format	output format (Default: BIOPAX). Valid options can be found using pcFormats
verbose	a boolean, display the command used to query Pathway Commons

Details

Get commands only retrieve the BioPAX elements that are directly mapped to the ID. Use the "traverse query to traverse BioPAX graph and obtain child/owner elements.

Information on MIRIAM and Identifiers.org http://www.pathwaycommons.org/pc2/#miriam

Value

a XMLInternalDocument object

See Also

pcFormats

Examples

```
uri <- "http://identifiers.org/uniprot/O14503"
#results <- getPc(uri)

uri <- c("http://identifiers.org/uniprot/O14503", "http://identifiers.org/uniprot/Q9P2X7")
#results <- getPc(uri, verbose=TRUE)
```

---

getPcUrl *Get base Pathway Commons URL*

Description

Get base Pathway Commons URL

Usage

```
getPcUrl()
```

Details

paxtoolsr will support versions Pathway Commons 5 and later. Old versions of the webservice will not be not be operational. Users can parse older BioPAX outputs as an alternative.

Value

a string with base Pathway Commons URL

Examples

```
url <- getPcUrl()
```

`getShortestPathSif` *Get the shortest between two IDs (HGNC or CHEBI)*

Description

Get the shortest between two IDs (HGNC or CHEBI)

Usage

```
getShortestPathSif(sif, idA, idB, mode = c("all", "out", "in"),
  weights = NULL, filterFun, ...)
```

Arguments

`sif`	a SIF network
`idA`	HGNC or CHEBI (CHEBI:XXXXX) ID
`idB`	HGNC or CHEBI (CHEBI:XXXXX) ID
`mode`	see shortest_paths() in igraph
`weights`	see shortest_paths() in igraph
`filterFun`	a function to filter multiple paths of the same length
`...`	additional arguments passed on to filterFun

Value

a data.frame representing a SIF network

`getSifInteractionCategories`
Get a list of categories of SIF interactions

Description

Get a list of categories of SIF interactions

Usage

```
getSifInteractionCategories()
```

Details

Description of interaction types: http://www.pathwaycommons.org/pc2/formats Categories provided: BetweenProteins, BetweenProteinsOther (often from high-throughput experiments), BetweenProteinSmallMolecule, BetweenSmallMolecules, SignedInteractions

Value

a list of interactions in categories

Examples

```
sifCat <- getSifInteractionCategories()
sifCat["BetweenProteins"]]
```

---

graphPc *Get Pathway Commons BioPAX elements*

Description

This function will retrieve a set of BioPAX elements given a graph query match.

Usage

```
graphPc(kind, source, target = NULL, direction = NULL, limit = NULL,
  format = NULL, datasource = NULL, organism = NULL, verbose = FALSE)
```

Arguments

kind	graph query. Valid options can be found using pcGraphQueries See Details for information on graph queries.
source	source object's URI/ID. Multiple source URIs/IDs are allowed per query, for example c("http://identifiers.org/uniprot/Q06609", "http://identifiers.org/uniprot/Q549Z0") See a note about MIRIAM and Identifiers.org in details
target	[Required for PATHSFROMTO graph query] target URI/ID. Multiple target URIs are allowed per query; for example c("http://identifiers.org/uniprot/Q06609", "http://identifiers.org/uniprot/Q549Z0") See a note about MIRIAM and Identifiers.org in details
direction	[Optional, for NEIGHBORHOOD and COMMONSTREAM algorithms] - graph search direction. Valid options: pcDirections.
limit	graph query search distance limit (default: 1).
format	output format. Valid options: pcFormats
datasource	datasource filter (same as for 'search').
organism	organism filter (same as for 'search').
verbose	a boolean, display the command used to query Pathway Commons

Value

depending on the the output format a different object may be returned. pcFormats

See Also

pcFormats, pcDirections

Examples

```
source <- "http://identifiers.org/uniprot/O14503"
#results <- graphPc(source=source, kind="neighborhood", format="EXTENDED_BINARY_SIF")
```

`idMapping` *Map IDs to Primary Uniprot or ChEBI IDs*

Description

Unambiguously maps, e.g., HGNC gene symbols, NCBI Gene, RefSeq, ENS*, and secondary UniProt identifiers to the primary UniProt accessions, or - ChEBI and PubChem IDs to primary ChEBI. You can mix different standard ID types in one query.

Usage

```
idMapping(ids, verbose = FALSE)
```

Arguments

`ids`	a vector of IDs
`verbose`	a boolean, display the command used to query Pathway Commons

Details

This is a specific id-mapping (not general-purpose) for reference proteins and small molecules; it was first designed for internal use, such as to improve BioPAX data integration and allow for graph queries accept not only URIs but also standard IDs. The mapping tables were derived exclusively from Swiss-Prot (DR fields) and ChEBI data (manually created tables and other mapping types and sources can be added in the future versions if necessary).

Value

a list of where each entry is a HGNC symbol provided and the each value is a primary UniProt or ChEBI ID.

Examples

```
genes <- c("BRCA2", "TP53")
#results <- idMapping(genes)
```

integrateBiopax *Integrate two BioPAX OWL files (DEPRECATED)*

Description

This function merges two BioPAX OWL files

Usage

```
integrateBiopax(inputFile1, inputFile2, outputFile = NULL)
```

Arguments

inputFile1	a string of the name of the input BioPAX OWL file
inputFile2	a string of the name of the input BioPAX OWL file
outputFile	a string of the name of the output integrated BioPAX OWL file

Details

This method is deprecated. Use mergeBiopax instead.

Value

an XMLInternalDocument representing a BioPAX OWL file

See Also

mergeBiopax

Examples

```
outFile <- tempfile()
results <- integrateBiopax(system.file("extdata", "raf_map_kinase_cascade_reactome.owl",
  package="paxtoolsr"),
  system.file("extdata", "dna_replication.owl", package="paxtoolsr"),
  outFile)
```

loadSifInIgraph *Load SIF as igraph Network*

Description

Load SIF as igraph Network

Usage

```
loadSifInIgraph(sif, directed = TRUE)
```

Arguments

sif	a binary SIF as a data.frame with three columns: "PARTICIPANT_A", "INTERACTION_TYPE", "PARTICIPANT_B"
directed	a boolean weather the returned graph should be directed (DEFAULT: TRUE)

Details

Users are likely to run into issues if the input SIF has factor levels

Value

a directed igraph network with interaction types

Examples

```
results <- readSif(system.file("extdata", "test_sif.txt", package="paxtoolsr"))
g <- loadSifInIgraph(results)
```

---

mapValues *Map values from One Vector to Another*

Description

Map values from One Vector to Another

Usage

```
mapValues(data, oldValue, newValue)
```

Arguments

data	a vector of strings where values will be replaced
oldValue	a vector that matches values in the data vector
newValue	a vector of new values that will replace the old values

Value

return the vector with the mapped values. If there was no corresponding entry then replace it with an NA.

Examples

```
data <- c("A", "B", "C", "X", "Y", "Z")
oldValue <- LETTERS[1:20]
newValue <- letters[1:20]
results <- mapValues(data, oldValue, newValue)
```

mergeBiopax *Merges two BioPAX OWL files*

Description

This function merges two BioPAX OWL files

Usage

```
mergeBiopax(inputFile1, inputFile2, outputFile = NULL)
```

Arguments

inputFile1	a string of the name of the input BioPAX OWL file
inputFile2	a string of the name of the input BioPAX OWL file
outputFile	a string of the name of the output merged BioPAX OWL file (Optional)

Details

Only entities that share IDs will be merged. No additional merging occurs on cross-references. Merging may result in warning messages caused as a result of redundant actions being checked against by the Java library; these messages may be ignored.

Value

an XMLInternalDocument representing a BioPAX OWL file

Examples

```
outFile <- tempfile()
results <- mergeBiopax(system.file("extdata", "raf_map_kinase_cascade_reactome.owl",
                       package="paxtoolsr"),
                       system.file("extdata", "dna_replication.owl",
                       package="paxtoolsr"),
                       outFile)
```

pcDirections *Acceptable Pathway Commons Directions*

Description

A simple function to see valid options

Usage

```
pcDirections()
```

Details

- BOTHSTREAM where the current entity can either be the source or target of an interaction
- DOWNSTREAM where the current entity can only be the source
- UPSTREAM where the current entity can only be the target

Value

acceptable Pathway Commons directions

Examples

```
pcDirections()
```

---

pcFormats *Acceptable Pathway Commons Formats*

Description

A simple function to see valid options

Usage

```
pcFormats()
```

Details

See references.

Value

acceptable Pathway Commons formats

References

Output Formats Description: http://www.pathwaycommons.org/pc2/help/formats.html

Examples

```
pcFormats()
```

pcGraphQueries *Acceptable Pathway Commons Graph Queries*

Description

A simple function to see valid options

Usage

```
pcGraphQueries()
```

Details

- COMMONSTREAM searches common downstream or common upstream of a specified set of entities based on the given directions within the boundaries of a specified length limit
- NEIGHBORHOOD searches the neighborhood of given source set of nodes
- PATHSBETWEEN finds the paths between specific source set of states or entities within the boundaries of a specified length limit
- PATHSFROMTO finds the paths from a specific source set of states or entities to a specific target set of states or entities within the boundaries of a specified length limit

Value

acceptable Pathway Commons graph queries

Examples

```
pcGraphQueries()
```

processPcRequest *Process Pathway Commons request in various formats*

Description

Process Pathway Commons request in various formats

Usage

```
processPcRequest(content, format)
```

Arguments

content	a string, content to be processed
format	a string, the type of format

Value

an R object using one of the read* methods provided in this package corresponding to the format

See Also

pcFormats

Examples

```
fileName <- system.file("extdata", "test_biopax.owl", package="paxtoolsr")
content <- readChar(fileName, file.info(fileName)$size)
results <- processPcRequest(content, "BIOPAX")
```

---

readBiopax *Read BioPAX files as XML documents*

Description

Read BioPAX files as XML documents

Usage

```
readBiopax(inputFile)
```

Arguments

inputFile	an inputFile

Value

an XMLInternalDocument

Examples

```
results <- readBiopax(system.file("extdata", "biopax3-short-metabolic-pathway.owl",
  package="paxtoolsr"))
```

---

readGmt *Read in gene sets from GMT files*

Description

This function will read in gene sets in the GMT format into a named list.

Usage

```
readGmt(inputFile)
```

Arguments

inputFile	an inputFile

Value

a named list where each entry corresponds to a gene set

Examples

```
results <- readGmt(system.file("extdata", "test_gsea.gmt", package="paxtoolsr"))
```

`readSbgn` *Read SBGN files as XML documents*

Description

Read SBGN files as XML documents

Usage

```
readSbgn(inputFile)
```

Arguments

`inputFile`	an inputFile

Value

an XMLInternalDocument

Examples

```
results <- readSbgn(system.file("extdata", "test_sbgn.xml", package="paxtoolsr"))
```

`readSif` *Read in a binary SIF file*

Description

Read in a binary SIF file

Usage

```
readSif(inputFile)
```

Arguments

`inputFile`	an inputFile

Value

a data.frame with the interactions in the binary SIF format

Examples

```
results <- readSif(system.file("extdata", "test_sif.txt", package="paxtoolsr"))
```

---

readSifnx *Read in a Extended SIF file*

Description

Read in a Extended SIF file

Usage

```
readSifnx(inputFile, asDT = TRUE)
```

Arguments

inputFile	an inputFile
asDT	TODO

Details

SIFNX files from Pathway Commons commonly come a single file that includes a tab-delimited sections for nodes and another for edges. The sections are separated by an empty lines. These sections must be split before they are read.

Value

a list with nodes and edges entries

Examples

```
results <- readSifnx(system.file("extdata", "test_sifnx.txt", package="paxtoolsr"))
chebiIds <- lapply(results$nodesUniXref, function(x) { x[which(grepl("CHEBI", x))] })
```

---

searchListOfVectors *Search List of Vectors*

Description

Search List of Vectors

Usage

```
searchListOfVectors(q, lst)
```

Arguments

q	query vector
lst	list of vectors to search

Details

Taken from: http://stackoverflow.com/questions/11002391/fast-way-of-getting-index-of-match-in-list

Value

a list of vectors with the same length as the query vector, each list entry will have indicies for lst where there was a match with the query vector. Return NA if there were no matches.

Examples

```
lst <- list(1:3, 3:5, 3:7)
q <- c(3, 5)
results <- searchListOfVectors(q, lst)
names(results) <- q

lst <- list(LETTERS[1:3], LETTERS[3:5], LETTERS[3:7])
q <- c("C", "E")
searchListOfVectors(q, lst)

lst <- list(LETTERS[3], LETTERS[4:6])
q <- "C"
searchListOfVectors(q, lst)

lst <- list(LETTERS[3], LETTERS[4:6])
q <- c("C")
searchListOfVectors(q, lst)

lst <- list(LETTERS[3], LETTERS[4:6])
q <- c("C", "E")
searchListOfVectors(q, lst)

lst <- list(LETTERS[3], LETTERS[4:6])
q <- "Z"
searchListOfVectors(q, lst)
```

---

searchPc *Search Pathway Commons*

Description

This command provides a text search using the Lucene query syntax.

Usage

```
searchPc(q, page = 0, datasource = NULL, organism = NULL, type = NULL,
  verbose = FALSE)
```

Arguments

q	a keyword, name, external identifier, or a Lucene query string.
page	an integer giving the search result page number (N>=0, default: 0)
datasource	a vector that is a filter by data source (use names or URIs of pathway data sources or of any existing Provenance object). If multiple data source values are specified, a union of hits from specified sources is returned. For example, datasource as c("reactome", "pid") returns hits associated with Reactome or PID.
organism	a vector that is an organism filter. The organism can be specified either by official name, e.g. "homo sapiens" or by NCBI taxonomy id, e.g. "9606". Similar to data sources, if multiple organisms are declared a union of all hits from specified organisms is returned. For example organism as c("9606", "10016") returns results for both human and mice. Only humans, "9606" is officially supported.
type	BioPAX class filter. See Details.
verbose	a boolean, display the command used to query Pathway Commons

Details

Indexed fields were selected based on most common searches. Some of these fields are direct BioPAX properties, others are composite relationships. All index fields are (case-sensitive):comment, ecnumber, keyword, name, pathway, term, xrefdb, xrefid, dataSource, and organism. The pathway field maps to all participants of pathways that contain the keyword(s) in any of its text fields. This field is transitive in the sense that participants of all sub-pathways are also returned. Finally, keyword is a transitive aggregate field that includes all searchable keywords of that element and its child elements - e.g. a complex would be returned by a keyword search if one of its members has a match. Keyword is the default field type. All searches can also be filtered by data source and organism. It is also possible to restrict the domain class using the 'type' parameter. This query can be used standalone or to retrieve starting points for graph searches. Search strings are case insensitive unless put inside quotes.

BioPAX classes can be found at http://www.pathwaycommons.org/pc2/#biopax_types

Value

an XMLInternalDocument with results

Examples

```
query <- "Q06609"
#results <- searchPc(query)

query <- "glycolysis"
#results <- searchPc(query, type="Pathway")
```

---

skip_on_bioc *Skip a test if on Bioconductor*

Description

Extension on testthat code

Usage

```
skip_on_bioc()
```

Value

A boolean or NULL is returned

---

splitSifnxByPathway *Splits SIFNX entries into individual pathways*

Description

Splits SIFNX entries into individual pathways

Usage

```
splitSifnxByPathway(edges, parallel = FALSE)
```

Arguments

edges	a data.frame with SIF content with the additional column "PATHWAY_NAMES". "PATHWAY_NAMES" should include pathway names delimited with a semi-colon: ";".
parallel	a boolean that will parallelize the process; requires foreach/doSNOW/parallel packages

Details

This method can be slow; ~1.5 minutes for 150K+ rows. Has a parallelized method to speed things up.

Value

a list of where each entry is a vector of row indicies for a given pathway

---

summarize *Summarize a BioPAX file*

Description

This function provides a summary of BioPAX classes.

Usage

```
summarize(inputFile)
```

Arguments

inputFile	a string of the name of the input BioPAX OWL file

Details

BioPAX classes are defined by the BioPAX specification: http://www.biopax.org/

Value

list with BioPAX class counts

Examples

```
summary <- summarize(system.file("extdata", "raf_map_kinase_cascade_reactome.owl",
package="paxtoolsr"))
```

---

summarizeSif *Summarize a SIF Network*

Description

Summarize a SIF Network

Usage

```
summarizeSif(sif)
```

Arguments

sif	a binary SIF as a data.frame with three columns: "PARTICIPANT_A", "INTERACTION_TYPE", "PARTICIPANT_B"

Value

a list containing a count of the unique genes in the SIF and counts for the interaction types in the network

Examples

```
results <- readSif(system.file("extdata", "test_sif.txt", package="paxtoolsr"))
summarizeSif(results)
```

toGSEA *Converts a BioPAX OWL file to a GSEA GMT gene set*

Description

This function converts pathway information stored as BioPAX files into the the GSEA .gmt format.

Usage

```
toGSEA(inputFile, outputFile = NULL, database, crossSpeciesCheckFlag)
```

Arguments

inputFile	a string of the name of the input OWL file
outputFile	a string of the name of the output file
database	a string of the name of the identifier type to be included (e.g. "HGNC Symbol")
crossSpeciesCheckFlag	a boolean that ensures participant protein is from same species

Details

The GSEA GMT format is a tab-delimited format where each row represents a gene set. The first column is the gene set name. The second column is a brief description. Other columns for each row contain genes in the gene set; these rows may be of unequal lengths.

Value

see readGmt()

Examples

```
outFile <- tempfile()
results <- toGSEA(system.file("extdata", "biopax3-short-metabolic-pathway.owl",
                              package="paxtoolsr"),
                              outFile,
                              "uniprot",
                              crossSpeciesCheckFlag=TRUE)
```

toLevel3 *Convert a PSIMI or older BioPAX OWL file to BioPAX Level 3*

Description

This file will convert PSIMI or older BioPAX objects to BioPAX Level 3

Usage

```
toLevel3(inputFile, outputFile = NULL)
```

Arguments

inputFile	a string of the name of the input file
outputFile	a string of the name of the output BioPAX OWL file

Value

an XMLInternalDocument representing a BioPAX OWL file

Examples

```
inputFile <- system.file("extdata", "raf_map_kinase_cascade_reactome.owl",
  package="paxtoolsr")
outFile <- tempfile()
results <- toLevel3(inputFile, outFile)
```

---

topPathways *Retrieve top pathways*

Description

This command returns all "top" pathways.

Usage

```
topPathways(datasource = NULL, organism = NULL, verbose = FALSE)
```

Arguments

datasource	filter by data source (same as for 'search').
organism	organism filter (same as for 'search').
verbose	a boolean, display the command used to query Pathway Commons

Details

Pathways that are neither 'controlled' nor 'pathwayComponent' of another process.

Value

a data.frame with the following columns:

- uri URI ID for the pathway
- biopaxClass the type of BioPAX object
- name a human readable name
- dataSource the dataSource for the pathway
- organism an organism identifier
- pathway URI ID for the pathway

Examples

```
datasource <- "panther"
#results <- topPathways(datasource=datasource)
```

toSBGN *Convert a BioPAX OWL file to SBGNML*

Description

This function will convert a BioPAX OWL file into the Systems Biology Graphical Notation (SBGN) Markup Language (SBGNML) XML representation

Usage

```
toSBGN(inputFile, outputFile = NULL)
```

Arguments

inputFile	a string of the name of the input BioPAX OWL file
outputFile	a string of the name of the output SBGNML file

Details

Objects in the SBGNML format are laid out using a Compound Spring Embedder (CoSE) layout

Value

see readSbgn()

References

http://www.cs.bilkent.edu.tr/~ivis/layout/cose-animated-demo/cose.html

Examples

```
outFile <- tempfile()
results <- toSBGN(system.file("extdata", "biopax3-short-metabolic-pathway.owl",
  package="paxtoolsr"),
  outFile)
```

toSif *Convert a BioPAX OWL file to SIF*

Description

Convert a BioPAX OWL file to a binary SIF file

Usage

```
toSif(inputFile, outputFile = NULL)
```

Arguments

inputFile	a string of the name of the input BioPAX OWL file
outputFile	a string of the name of the output SIF file (Optional)

Details

Information on SIF conversion is provided on the Pathway Commons site: http://www.pathwaycommons.org/pc2/

Value

see readSif()

Examples

```
outFile <- tempfile()
results <- toSif(system.file("extdata", "raf_map_kinase_cascade_reactome.owl",
  package="paxtoolsr"),
  outFile)
```

---

toSifnx *Converts BioPAX OWL file to extended binary SIF representation*

Description

Converts BioPAX OWL file to extended binary SIF representation

Usage

```
toSifnx(inputFile, outputFile = tempfile())
```

Arguments

inputFile	a string with the name of the input BioPAX OWL file
outputFile	a string with the name of the output file for SIFNX information

Details

Information on SIF conversion is provided on the Pathway Commons site: http://www.pathwaycommons.org/pc2/

Value

see readSifnx()

Examples

```
inputFile <- system.file("extdata", "raf_map_kinase_cascade_reactome.owl", package="paxtoolsr")
results <- toSifnx(inputFile=inputFile)
```

traverse *Access Pathway Commons using XPath-type expressions*

Description

This command provides XPath-like access to the Pathway Commons.

Usage

```
traverse(uri, path, verbose = FALSE)
```

Arguments

uri	a BioPAX element URI - specified similarly to the 'GET' command above). Multiple IDs are allowed (uri=...&uri=...&uri=...).
path	a BioPAX propery path in the form of property1[:type1]/property2[:type2]; see properties, inverse properties, Paxtools, org.biopax.paxtools.controller.PathAccessor.
verbose	a boolean, display the command used to query Pathway Commons

Details

With traverse users can explicitly state the paths they would like to access. The format of the path query is in the form: [Initial Class]/[property1]:[classRestriction(optional)]/[property2]... A "*" sign after the property instructs path accessor to transitively traverse that property. For example, the following path accessor will traverse through all physical entity components within a complex: "Complex/component*/entityReference/xref:UnificationXref" The following will list display names of all participants of interactions, which are components (pathwayComponent) of a pathway (note: pathwayOrder property, where same or other interactions can be reached, is not considered here): "Pathway/pathwayComponent:Interaction/participant*/displayName" The optional parameter classRestriction allows to restrict/filter the returned property values to a certain subclass of the range of that property. In the first example above, this is used to get only the Unification Xrefs. Path accessors can use all the official BioPAX properties as well as additional derived classes and parameters in paxtools such as inverse parameters and interfaces that represent anonymous union classes in OWL. (See Paxtools documentation for more details).

Value

an XMLInternalDocument with results

References

Paxtools Documentation: http://www.biopax.org/m2site/

Examples

```
uri <- "http://identifiers.org/uniprot/P38398"
#results <- traverse(uri=uri, path="ProteinReference/organism/displayName")
```

validate *Validate BioPAX files*

Description

This function validates BioPAX files for errors.

Usage

```
validate(inputFile, outputFile = NULL, type = c("xml", "html", "biopax"),
  autoFix = FALSE, onlyErrors = FALSE, maxErrors = NULL,
  notStrict = FALSE)
```

Arguments

inputFile	a string of the name of the input BioPAX OWL file
outputFile	a string of the name of the output file containing validation results
type	a string denoting the type of output: xml (default), html, biopax
autoFix	a boolean that determines if the input file should be fixed automatically. Errors that can be automatically fixed include generating displayName properties from names, inferring organism, and inferring dataSource
onlyErrors	a boolean of whether to only display errors
maxErrors	a integer denoting the number of errors to return
notStrict	a boolean of whether to be strict in validation (default: FALSE)

Details

See the publication by Rodchenkov, et al. for information on the BioPAX validator. See http://biopax.baderlab.org/validator for additional information on validator. See http://biopax.baderlab.org/validator/errorTypes.html for information on error types.

Value

an XMLInternalDocument is returned if type is set to "xml" otherwise the location of the outputfile is returned.

References

Rodchenkov I, Demir E, Sander C, Bader GD. The BioPAX Validator, http://www.ncbi.nlm.nih.gov/pubmed/23918249

Examples

```
outFile <- tempfile()
rawDoc <- validate(system.file("extdata", "raf_map_kinase_cascade_reactome.owl",
  package="paxtoolsr"), onlyErrors=TRUE)
```

Index