

# Package ‘drugTargetInteractions’

November 20, 2024

**Type** Package

**Title** Drug-Target Interactions

**Version** 1.14.0

**Date** 2023-10-24

**Description** Provides utilities for identifying drug-target interactions for sets of small molecule or gene/protein identifiers. The required drug-target interaction information is obtained from a local SQLite instance of the ChEMBL database. ChEMBL has been chosen for this purpose, because it provides one of the most comprehensive and best annotated knowledge resources for drug-target information available in the public domain.

**Depends** methods, R (>= 4.1)

**Imports** utils, RSQLite, UniProt.ws, biomaRt,ensembldb,  
BiocFileCache,dplyr,rappdirs, AnnotationFilter, S4Vectors

**Suggests** RUnit, BiocStyle, knitr, rmarkdown, ggplot2, reshape2, DT,  
EnsDb.Hsapiens.v86

**VignetteBuilder** knitr

**License** Artistic-2.0

**NeedsCompilation** no

**URL** <https://github.com/girke-lab/drugTargetInteractions>

**biocViews** Cheminformatics, BiomedicalInformatics, Pharmacogenetics,  
Pharmacogenomics, Proteomics, Metabolomics

**RoxygenNote** 7.1.1

**BugReports** <https://github.com/girke-lab/drugTargetInteractions>

**git\_url** <https://git.bioconductor.org/packages/drugTargetInteractions>

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** 09f273e

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2024-11-19

**Author** Thomas Girke [cre, aut]

**Maintainer** Thomas Girke <[thomas.girke@ucr.edu](mailto:thomas.girke@ucr.edu)>

## Contents

drugTargetInteractions-package . . . . .	2
cmpIdMapping . . . . .	3
downloadChEMBLdb . . . . .	4
downloadUniChem . . . . .	4
drugTargetAnnot . . . . .	5
drugTargetAnnotTable . . . . .	6
drugTargetBioactivity . . . . .	7
genConfig . . . . .	8
getDrugTarget . . . . .	8
getParalogs . . . . .	9
getSymEnsUp . . . . .	10
getUniprotIDs . . . . .	11
processDrugage . . . . .	12
runDrugTarget_Annot_Bioassay . . . . .	12
transformTTD . . . . .	14
<b>Index</b>	<b>15</b>

---

drugTargetInteractions-package  
*Drug-Target Interactions*

---

## Description

The drugTargetInteractions package provides utilities for identifying drug-target interactions for sets of small molecule or gene/protein identifiers. The required drug-target interaction information is obtained from a local SQLite instance of the ChEMBL database.

## Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

## Author(s)

Thomas Girke [cre, aut]

Maintainer: Thomas Girke <thomas.girke@ucr.edu>

---

cmpIdMapping	<i>cmpIdMapping</i>
--------------	---------------------

---

## Description

Function to generate compound ID mappings UniChem.

This function requires the ID mapping files "src1src2.txt.gz", "src1src22.txt.gz", and "src1src7.txt.gz" to exist in a directory called "downloads" before being run. These can be generated with the [downloadUniChem](#) function.

It will do some processing on these files and output an RDS file at outfile. This file can then be used in other functions, such as [drugTargetAnnot](#).

## Usage

```
cmpIdMapping(outfile=file.path(config$resultsPath,"cmp_ids.rds"), rerun=TRUE,config=genConfig())
```

## Arguments

outfile	Path to output file.
rerun	If true, runs processing, otherwise does nothing.
config	General configuration. See <a href="#">genConfig</a> .

## Value

Generates an RDS file at outfile.

## Author(s)

Thomas Girke

## See Also

[downloadUniChem](#)

## Examples

```
cmpIdMapping("cmp_ids.rds",rerun=FALSE)
```

---

downloadChemblDb	<i>downloadChemblDb</i>
------------------	-------------------------

---

**Description**

Download ChEMBL sqlite db for use by several other functions in the package.

**Usage**

```
downloadChemblDb(version, rerun=TRUE, config=genConfig())
```

**Arguments**

version	The ChEMBL version to download.
rerun	If TRUE, the file will be downloaded, otherwise do nothing.
config	The configuration object. This gives the location to put the downloaded chembl db.

**Value**

No return value.

**Author(s)**

Kevin Horan

**Examples**

```
downloadChemblDb(27)
```

---

downloadUniChem	<i>downloadUniChem</i>
-----------------	------------------------

---

**Description**

Downloads UniChem compound ID mappings from <https://www.ebi.ac.uk/unicem/ucquery/listSources>. Mappings are downloaded for DrugBank, PubChem, and ChEBI.

**Usage**

```
downloadUniChem(rerun=TRUE, config=genConfig())
```

**Arguments**

rerun	If true, downloads the files, else does nothing.
config	General configuration. See <a href="#">genConfig</a> .

**Value**

Generates the following output files: "src1src2.txt.gz", "src1src22.txt.gz", and "src1src7.txt.gz". These correspond to mappings from ChEMBL to DrugBank, PubChem, and ChEBI, respectively.

**Author(s)**

Thomas Girke

**References**

<https://www.ebi.ac.uk/unichem/ucquery/listSources>

**See Also**

[drugTargetAnnotTable](#)

**Examples**

```
downloadUniChem(rerun=TRUE)
```

---

drugTargetAnnot

*drugTargetAnnot*

---

**Description**

Function to query known drug-target annotations.

**Usage**

```
drugTargetAnnot(queryBy=list(molType=NULL, idType=NULL, ids=NULL), cmpid_file=file.path(config$re
```

**Arguments**

queryBy	A list defining the query, as described in <a href="#">queryBy</a> .
cmpid_file	Path to a compound ID mapping file, generated by <a href="#">cmpIdMapping</a> .
config	General configuration. See <a href="#">genConfig</a> .

**Value**

Returns the query results as a data frame.

**Author(s)**

Thomas Girke

**See Also**

[queryBy](#) [cmpIdMapping](#)

## Examples

```
# These are just sample files included in the package.
# You should use your own data files.
config = genConfig(chemblDbPath=
  system.file("extdata", "chembl_sample.db", package="drugTargetInteractions"),
  resultsPath =
  system.file("extdata", "results", package="drugTargetInteractions"))

queryBy <- list(molType="cmp", idType="chembl_id", ids=c("CHEMBL1233058", "CHEMBL1200916", "CHEMBL437765"))
qresult <- drugTargetAnnot(queryBy, config=config)
```

---

drugTargetAnnotTable *drugTargetAnnotTable*

---

## Description

Generates a drug target annotation TSV file. This file includes target information from ChEMBL, drugbank, pubchem, and chembi.

This function requires the ID mapping files "src1src2.txt.gz", "src1src22.txt.gz", and "src1src7.txt.gz" to exist in a directory called "downloads" before being run. These can be generated with the [downloadUniChem](#) function.

## Usage

```
drugTargetAnnotTable(outfile, rerun=TRUE, config=genConfig())
```

## Arguments

outfile	The name of the output file to write the results to.
rerun	If true, download and generate output file. Otherwise do nothing.
config	General configuration. See <a href="#">genConfig</a> .

## Value

Writes output file to outfile.

## Author(s)

Thomas Girke

## See Also

[downloadUniChem](#)

## Examples

```
config = genConfig(chemblDbPath=
  system.file("extdata", "chembl_sample.db", package="drugTargetInteractions"))
drugTargetAnnotTable(outfile="drugTargetAnnot.xls", config=config)
```

---

drugTargetBioactivity *drugTargetBioactivity*

---

## Description

Function to query bioactivity data by target or compound ids

## Usage

```
drugTargetBioactivity( queryBy=list(molType=NULL, idType=NULL, ids=NULL), cmpid_file=file.path(co
```

## Arguments

queryBy	A list defining the query, as described in <a href="#">queryBy</a> .
cmpid_file	Path to a compound ID mapping file, generated by <a href="#">cmpIdMapping</a> .
config	General configuration. See <a href="#">genConfig</a> .

## Value

Returns results as a data frame.

## Author(s)

Thomas Girke

## See Also

[queryBy](#) [cmpIdMapping](#)

## Examples

```
config = genConfig(chemblDbPath=
  system.file("extdata", "chembl_sample.db", package="drugTargetInteractions"),
  resultsPath =
  system.file("extdata", "results", package="drugTargetInteractions"))
```

```
queryBy <- list(molType="protein", idType="uniprot", ids=c("P05979", "P35354", "P33033", "Q8VCT3", "P29475"))
qresult <- drugTargetBioactivity( queryBy, config=config)
```

```
queryBy <- list(molType="cmp", idType="molregno", ids=c("101036", "101137", "1384464"))
qresult <- drugTargetBioactivity( queryBy, config=config)
```

```
queryBy <- list(molType="cmp", idType="DrugBank_ID", ids=c("DB00945", "DB00316", "DB01050"))
qresult <- drugTargetBioactivity(queryBy, config=config)
```

```
queryBy <- list(molType="cmp", idType="PubChem_ID", ids=c("2244", "3672", "1983"))
qresult <- drugTargetBioactivity(queryBy, config=config)
```

---

genConfig	<i>genConfig</i>
-----------	------------------

---

### Description

Create a default configuration object.

### Usage

```
genConfig( chemblDbPath = "chembl.db", downloadPath = "downloads", resultsPath = "results" )
```

### Arguments

chemblDbPath	Path or filename of ChEMBL SQLite db file.
downloadPath	The name of a directory to put downloaded files in.
resultsPath	The name of a directory to put output files in.

### Value

A config object that can be passed to their functions.

### Author(s)

Kevin Horan

### Examples

```
config = genConfig()
```

---

getDrugTarget	<i>getDrugTarget</i>
---------------	----------------------

---

### Description

This function allows you to query a subset of the data fetched by [drugTargetAnnotTable](#).

### Usage

```
getDrugTarget(dt_file=file.path(config$resultsPath, "drugTargetAnnot.xls"), queryBy=list(molType=
  id_mapping=c(chembl="chembl_id", pubchem="PubChem_ID", uniprot="UniProt_ID"), columns,conf
```

### Arguments

dt_file	The drug target annotation file. This can be generated with <a href="#">drugTargetAnnotTable</a> .
queryBy	A list defining the query, as described in <a href="#">queryBy</a> .
id_mapping	A list providing the id columns for ChEMBL, PubChem, and UniProt. It should contain the fields "chembl", "pubchem", and "uniprot", each with the column name of the respective id number in the drug target annotation file. See default value above for an example.
columns	A list of column indexes to select as a subset of the final result set.
config	General configuration. See <a href="#">genConfig</a> .



**Value**

Returns the query result as a data frame.

**Author(s)**

Thomas Girke

**See Also**

[drugTargetAnnotTable](#)

**Examples**

```
config = genConfig(chemblDbPath=
  system.file("extdata", "chembl_sample.db", package="drugTargetInteractions"),
  resultsPath =
  system.file("extdata", "results", package="drugTargetInteractions"))
id_mapping <- c(chembl="chembl_id", pubchem="PubChem_ID", uniprot="UniProt_ID", drugbank="DrugBank_ID")

queryBy <- list(molType="cmp", idType="chembl", ids=c("CHEMBL25", "CHEMBL1742471"))
getDrugTarget(queryBy=queryBy, id_mapping=id_mapping,
  columns=c(1,5,8,16,17),config=config)

queryBy <- list(molType="cmp", idType="pubchem", ids=c("2244", "65869", "2244"))
getDrugTarget(queryBy=queryBy, id_mapping=id_mapping,
  columns=c(1,5,8,16,17),config=config)

queryBy <- list(molType="protein", idType="uniprot", ids=c("P43166", "P00915", "P43166"))
getDrugTarget(queryBy=queryBy, id_mapping=id_mapping,
  columns=c(1,5,8,16,17),config=config)
```

---

getParalogs

*getParalogs*

---

**Description**

Using biomaRt, obtain for query genes the corresponding UniProt IDs as well as paralogs. Query genes can be Gene Names or ENSEMBL Gene IDs from H sapiens. The result is similar to IDMs and SSNNs from [getUniprotIDs](#) function, but instead of UNIREF clusters, biomaRt's paralogs are used to obtain SSNNs.

**Usage**

```
getParalogs(queryBy)
```

**Arguments**

queryBy            A list defining the query, as described in [queryBy](#).

**Value**

Returns a list with the paralogs for the given genes.

**Author(s)**

Thomas Girke

**See Also**[getUniprotIDs queryBy](#)**Examples**

```
queryBy <- list(molType="gene", idType="external_gene_name", ids=c("ZPBP", "MAPK1", "EGFR"))

#requires network connection and is slow
result <- getParalogs(queryBy)
```

getSymEnsUp

*Gene to Protein ID Mappings***Description**

The getSymEnsUp function returns for a query of gene or protein IDs a mapping table containing: ENSEMBL Gene IDs, Gene Names/Symbols, UniProt IDs and ENSEMBL Protein IDs. Subsequent slots contain the corresponding named character vectors. Internally, the function uses the ensemblDb package.

**Usage**

```
getSymEnsUp(EnsDb = "EnsDb.Hsapiens.v86", ids, idtype)
```

**Arguments**

EnsDb	EnsDb instance of ensemblDb package
ids	Character vector with IDs matching the type specified under idtype
idtype	Character vector of length one containing one of: GENE_NAME, ENSEMBL_GENE_ID or UNIPROT_ID

**Value**

List object with following components:

idDF	ID mapping data.frame
ens_gene_id	named character vector
up_ens_id	named character vector
up_gene_id	named character vector

**Author(s)**

Thomas Girke

**Examples**

```
gene_name <- c("CA7", "CFTR")
getSymEnsUp(EnsDb="EnsDb.Hsapiens.v86", ids=gene_name, idtype="GENE_NAME")
```

---

`getUniprotIDs`*Retrieve UniProt IDs via ID and Cluster Mappings*

---

### Description

The following returns for a set of query IDs (e.g. Ensembl gene IDs) the corresponding UniProt IDs based on two independent approaches: ID mappings (IDMs) and sequence similarity nearest neighbors (SSNNs) using UNIREF clusters. Note, the 'keys' or query IDs (e.g. ENSEMBL genes) can only be reliably maintained in the SSNN results when 'chunksize=1' since batch queries for protein clusters with 'UnitProt.ws' will often drop the query IDs. To address this, the query result contains an extra 'QueryID' column when 'chunksize=1', but not when it is set to a different value than 1.

The [getParalogs](#) function is similar but it uses biomaRt's paralogs instead of UNIREF clusters.

### Usage

```
getUniprotIDs(taxId = 9606, kt = "ENSEMBL", keys, seq_cluster = "UNIREF90", chunksize=20)
```

### Arguments

<code>taxId</code>	An NCBI taxonomy ID
<code>kt</code>	Should be either "ENSEMBL" or "UNIPROTKB".
<code>keys</code>	Query IDs.
<code>seq_cluster</code>	Which cluster to use. Should be one of 'UNIREF100', 'UNIREF90', 'UNIREF50'.
<code>chunksize</code>	Queries are done in batches, this parameter sets the size of each batch.

### Value

Returns a list of data.

### Author(s)

Thomas Girke

### See Also

[getParalogs UniProt.ws](#)

### Examples

```
keys <- c("ENSG00000145700", "ENSG00000135441", "ENSG00000120071", "ENSG00000120088", "ENSG00000185829", "E  
res_list100 <- getUniprotIDs(taxId=9606, kt="ENSEMBL", keys=keys, seq_cluster="UNIREF100")
```

---

processDrugage	<i>processDrugage</i>
----------------	-----------------------

---

**Description**

Download Drug Age data from [genomics.senescence.info/drugs](http://genomics.senescence.info/drugs). Process the data and write it out as a TSV spreadsheet.

**Usage**

```
processDrugage(drugagefile=file.path(config$resultsPath,"drugage_id_mapping.xls"), redownloaddrugage
```

**Arguments**

drugagefile	The name of the output file.
redownloaddrugage	If true, download the data file. Otherwise assume the file is already downloaded.
config	General configuration. See <a href="#">genConfig</a> .

**Value**

Output is written to drugagefile.

**Author(s)**

Thomas Girke

**Examples**

```
tryCatch({
  config = genConfig(chemblDbPath=
    system.file("extdata", "chembl_sample.db", package="drugTargetInteractions"))
  processDrugage("drugage_id_mapping.xls", TRUE, config)
},
error=function(e){
  message("Failed to run processDrugage(), please try again later")
})
```

---

runDrugTarget_Annot_Bioassay	<i>runDrugTarget_Annot_Bioassay</i>
------------------------------	-------------------------------------

---

**Description**

Meta-function to obtain in one step both drug-target annotation and bioassay data.

**Usage**

```
runDrugTarget_Annot_Bioassay(res_list, up_col_id="ID", ens_gene_id, cmpid_file=file.path(config$
```

**Arguments**

res_list	Object obtained from getUniprotIDs function.
up_col_id	Column name in data.frames of res_list containing uniprot IDs, usually one of: 'ID', 'ID_up_sp' or 'ID_up_sp_tr'.
ens_gene_id	Named character vector with ENSEMBL gene IDs in name slot and gene symbols or other ID type in value slot
cmpid_file	Path to CMP ID mapping file, often named cmp_ids.rds.
config	General configuration. See <a href="#">genConfig</a> .
...	Slot to pass on additional arguments.

**Value**

List with two components each containing a data.frame. The first one (Annotation) contains drug-target annotation data, and the second one (Bioassay) contains drug-target bioassay data.

**Author(s)**

Thomas Girke

**References**

References to be added...

**See Also**

See also: drugTargetAnnot and drugTargetBioactivity

**Examples**

```

config = genConfig(chemblDbPath=
  system.file("extdata", "chembl_sample.db", package="drugTargetInteractions"),
  resultsPath =
  system.file("extdata", "results", package="drugTargetInteractions"))

## (1) Translate gene symbols to ENSEMBL gene IDs
ensembl_gene_id <- c("ENSG0000001626", "ENSG00000168748")
idMap <- getSymEnsUp(EnsDb="EnsDb.Hsapiens.v86", ids=ensembl_gene_id, idtype="ENSEMBL_GENE_ID")
ens_gene_id <- idMap$ens_gene_id

## (2a) Retrieve UniProt IDs with both IDMs and SSNN paralogs
queryBy <- list(molType="gene", idType="ensembl_gene_id", ids=names(ens_gene_id))

#this function is slow and requires a network connection
res_list <- getParalogs(queryBy)

## (3) Obtain Drug-Target Annotation and Bioassay Data
drug_target_list <- runDrugTarget_Annot_Bioassay(res_list=res_list,
  up_col_id="ID_up_sp", ens_gene_id,config=config )

```

---

transformTTD	<i>transformTTD</i>
--------------	---------------------

---

**Description**

Integration with Therapeutic Target Database (TTD). This function downloads a data file from idr-blab.org and returns it as a data frame.

**Usage**

```
transformTTD(ttdfile=file.path(config$downloadPath, "TTD_IDs.txt"), redownloadTTD=TRUE, config=gen
```

**Arguments**

ttdfile	The name of the output file to write the downloaded file to.
redownloadTTD	If true, data file will be downloaded again. If false, we assume the file already exists at ttdfile.
config	General configuration. See <a href="#">genConfig</a> .

**Value**

Returns a data frame with TTD data in it.

**Author(s)**

Thomas Girke

**Examples**

```
ttd=tryCatch(  
  transformTTD(),  
  error=function(e){  
    message("Failed to download TTD file, please try again later")  
  }  
)
```

# Index

- \* **package**
  - drugTargetInteractions-package, [2](#)
- \* **utilities**
  - getSymEnsUp, [10](#)
  - runDrugTarget\_Annot\_Bioassay, [12](#)
- cmpIdMapping, [3](#), [5](#), [7](#)
- downloadChemblDb, [4](#)
- downloadUniChem, [3](#), [4](#), [6](#)
- drugTargetAnnot, [3](#), [5](#)
- drugTargetAnnotTable, [5](#), [6](#), [8](#), [9](#)
- drugTargetBioactivity, [7](#)
- drugTargetInteractions
  - (drugTargetInteractions-package), [2](#)
- drugTargetInteractions-package, [2](#)
- genConfig, [3–8](#), [8](#), [12–14](#)
- getDrugTarget, [8](#)
- getParalogs, [9](#), [11](#)
- getSymEnsUp, [10](#)
- getUniprotIDs, [9](#), [10](#), [11](#)
- processDrugage, [12](#)
- queryBy, [5](#), [7–10](#)
- queryBy (drugTargetAnnot), [5](#)
- runDrugTarget\_Annot\_Bioassay, [12](#)
- transformTTD, [14](#)
- UniProt.ws, [11](#)