# Package 'biodbUniprot'

May 28, 2025

**Title** biodbUniprot, a library for connecting to the Uniprot Database

**Version** 1.14.0

**Description** The biodbUniprot library is an extension of the biodb framework package. It provides access to the UniProt database. It allows to retrieve entries by their accession number, and run web service queries for searching for entries.

**URL** https://github.com/pkrog/biodbUniprot

**BugReports** https://github.com/pkrog/biodbUniprot/issues

**biocViews** Software, Infrastructure, DataImport

**Depends** R (>= 4.1.0)

**License** AGPL-3

**Encoding** UTF-8

**VignetteBuilder** knitr

**Suggests** BiocStyle, roxygen2, devtools, testthat (>= 2.0.0), knitr, rmarkdown, lgr, covr

**Imports** R6, biodb (>= 1.4.2)

**RoxygenNote** 7.2.1

**Collate** 'UniprotConn.R' 'UniprotEntry.R' 'package.R'

**git_url** https://git.bioconductor.org/packages/biodbUniprot

**git_branch** RELEASE_3_21

**git_last_commit** 9bc6bb0

**git_last_commit_date** 2025-04-15

**Repository** Bioconductor 3.21

**Date/Publication** 2025-05-28

**Author** Pierrick Roger [aut, cre] (ORCID: <https://orcid.org/0000-0001-8177-4873>)

**Maintainer** Pierrick Roger <pierrick.roger@cea.fr>

# Contents

UniprotConn . . . . . 2
UniprotEntry . . . . . 4

**Index** **6**

---

`UniprotConn` *The connector class to Uniprot database.*

---

## Description

The connector class to Uniprot database.

The connector class to Uniprot database.

## Details

This is a concrete connector class. It must never be instantiated directly, but instead be instantiated through the factory BiodbFactory. Only specific methods are described here. See super classes for the description of inherited methods.

## Super classes

`biodb::BiodbConnBase` -> `biodb::BiodbConn` -> `UniprotConn`

## Methods

### Public methods:

- `UniprotConn$wsSearch()`
- `UniprotConn$wsQuery()`
- `UniprotConn$geneSymbolToUniprotIds()`
- `UniprotConn$clone()`

**Method** `wsSearch()`**:** Calls search service on the database for searching for compounds. See https://www.uniprot.org/help/programmatic_access for details.

*Usage:*

```
UniprotConn$wsSearch(
  query = "",
  fields = NULL,
  format = NULL,
  size = NULL,
  retfmt = c("plain", "parsed", "ids", "request")
)
```

*Arguments:*

`query` The query to send to the database.

`fields` The field columns to retrieve from the database (e.g.: 'id', 'entry name', 'pathway', 'organism', 'sequence', etc).

`format` The return format (e.g.: 'tsv').

`size` The maximum number of entries to return.

`retfmt` Use to set the format of the returned value. 'plain' will return the raw results from the server, as a character value. 'parsed' will return the parsed results, as a JSON object. 'request' will return a BiodbRequest object representing the request as it would have been sent. 'ids' will return a character vector containing the IDs of the matching entries.

*Returns:* Depending on 'retfmt' parameter.

**Method** `wsQuery()`**:** Calls query service on the database for searching for compounds. See http //www.uniprot.org/help/api_queries for details.

*Usage:*

```
UniprotConn$wsQuery(
  query = "",
  columns = NULL,
  format = NULL,
  limit = NULL,
  retfmt = c("plain", "parsed", "ids", "request")
)
```

*Arguments:*

`query` The query to send to the database.

`columns` The field columns to retrieve from the database (e.g.: 'id', 'entry name', 'pathway', 'organism', 'sequence', etc).

`format` The return format (e.g.: 'tsv').

`limit` The maximum number of entries to return.

`retfmt` Use to set the format of the returned value. 'plain' will return the raw results from the server, as a character value. 'parsed' will return the parsed results, as a JSON object. 'request' will return a BiodbRequest object representing the request as it would have been sent. 'ids' will return a character vector containing the IDs of the matching entries.

*Returns:* Depending on 'retfmt' parameter.

**Method** `geneSymbolToUniprotIds()`**:** Gets UniProt IDs associated with gene symbols.

*Usage:*

```
UniprotConn$geneSymbolToUniprotIds(
  genes,
  ignore.nonalphanum = FALSE,
  partial.match = FALSE,
  filtering = TRUE,
  max.results = 0
)
```

*Arguments:*

`genes` A vector of gene symbols to convert to UniProt IDs.

`ignore.nonalphanum` If set to TRUE, do not take into account non-alphanumeric characters when comparing gene symbols.

`partial.match` If set to TRUE, a match will be valid even if the provided gene symbol is only a substring of the found gene symbol.

`filtering` If set to FALSE, do not run any filtering and return all the UniProt IDs given by UniProt search web service. DEPRECATED: new UniProt REST API returns only exact match.

`max.results` Maximum of UniProt IDs returned for each gene symbol.

*Returns:* A named list of vectors of UniProt IDs. The names are gene symbols provided with the genes parameter. For each gene symbol, a vector of found UniProt IDs is set.

**Method** `clone()`**:** The objects of this class are cloneable with this method.

*Usage:*

```
UniprotConn$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

## Examples

```
# Create an instance with default settings:
mybiodb <- biodb::newInst()

# Get Uniprot connector
uniprot <- mybiodb$getFactory()$createConn('uniprot')

# Access web service "search":
result <- uniprot$wsSearch(query='protein_name:"prion protein"',
                           fields=c('id', 'entry name'),
                           format='tsv', size=10)

# Terminate instance.
mybiodb$terminate()
```

---

`UniprotEntry` *Uniprot entry class.*

---

## Description

This is the entry class for Uniprot database.

## Super classes

`biodb::BiodbEntry` -> `biodb::BiodbXmlEntry` -> `UniprotEntry`

## Methods

### Public methods:

- `UniprotEntry$clone()`

**Method** `clone()`**:** The objects of this class are cloneable with this method.

*Usage:*

```
UniprotEntry$clone(deep = FALSE)
```

*Arguments:*

deep Whether to make a deep clone.

## Examples

```
# Create an instance with default settings:
mybiodb <- biodb::newInst()

# Create a connector
conn <- mybiodb$getFactory()$createConn('uniprot')

# Get an entry
e <- conn$getEntry('P01011')

# Terminate instance.
mybiodb$terminate()
```

# Index

biodb::BiodbConn, *2*
biodb::BiodbConnBase, *2*
biodb::BiodbEntry, *4*
biodb::BiodbXmlEntry, *4*

UniprotConn, 2
UniprotEntry, 4