

# Package ‘biodbExpasy’

May 1, 2024

**Title** biodbExpasy, a library for connecting to Expasy ENZYME database.

**Version** 1.9.0

**Description** The biodbExpasy library provides access to Expasy ENZYME database, using biodb package framework. It allows to retrieve entries by their accession number. Web services can be accessed for searching the database by name or comments.

**License** AGPL-3

**biocViews** Software, Infrastructure, DataImport

**VignetteBuilder** knitr

**Encoding** UTF-8

**Depends** R (>= 4.1)

**Imports** biodb (>= 1.3.1), R6, stringr, chk

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

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.2

**Collate** 'ExpasyEnzymeConn.R' 'ExpasyEnzymeEntry.R' 'package.R'

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

**git\_branch** devel

**git\_last\_commit** 25fc0fc

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.20

**Date/Publication** 2024-05-01

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

**Maintainer** Pierrick Roger <[pierrick.roger@cea.fr](mailto:pierrick.roger@cea.fr)>

## Contents

biodbExpasy-package . . . . .	2
ExpasyEnzymeConn . . . . .	2
ExpasyEnzymeEntry . . . . .	4

**Index****6**


---

biodbExpasy-package	<i>biodbExpasy: biodbExpasy, a library for connecting to Expasy ENZYME database.</i>
---------------------	--

---

**Description**

The biodbExpasy library provides access to Expasy ENZYME database, using biodb package framework. It allows to retrieve entries by their accession number. Web services can be accessed for searching the database by name or comments.

**Details**

See vignette biodbExpasy:

```
vignette('biodbExpasy', package='biodbExpasy')
```

**Author(s)**

**Maintainer:** Pierrick Roger <pierrick.roger@cea.fr> ([ORCID](#))

**See Also**

[ExpasyEnzymeConn](#).

---

ExpasyEnzymeConn	<i>Expasy ENZYME database. connector class.</i>
------------------	---

---

**Description**

Expasy ENZYME database. connector class.

Expasy ENZYME database. connector class.

**Details**

Connector class for Expasy ENZYME database.

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](#) -> ExpasyEnzymeConn

## Methods

### Public methods:

- [ExpasyEnzymeConn\\$new\(\)](#)
- [ExpasyEnzymeConn\\$wsEnzymeByName\(\)](#)
- [ExpasyEnzymeConn\\$wsEnzymeByComment\(\)](#)
- [ExpasyEnzymeConn\\$clone\(\)](#)

**Method** `new()`: New instance initializer. Connector classes must not be instantiated directly. Instead, you must use the `createConn()` method of the factory class.

*Usage:*

```
ExpasyEnzymeConn$new(...)
```

*Arguments:*

... All parameters are passed to the super class initializer.

*Returns:* Nothing.

**Method** `wsEnzymeByName()`: Calls enzyme-byname web service and returns the HTML result. See <http://enzyme.expasy.org/enzyme-byname.html>.

*Usage:*

```
ExpasyEnzymeConn$wsEnzymeByName(  
  name,  
  retfmt = c("plain", "request", "parsed", "ids")  
)
```

*Arguments:*

`name` The name to search for.

`retfmt` The format to use for the returned value. 'plain' will return the raw result from the server, as a character value. 'request' will return a `BiodbRequest` instance containing the request as it would have been sent. 'parsed' will return an XML object, containing the parsed result. 'ids' will return a character vector containing the IDs of the matching entries.

*Returns:* Depending on `retfmt`.

**Method** `wsEnzymeByComment()`: Calls enzyme-bycomment web service and returns the HTML result. See <http://enzyme.expasy.org/enzyme-bycomment.html>.

*Usage:*

```
ExpasyEnzymeConn$wsEnzymeByComment(  
  comment,  
  retfmt = c("plain", "request", "parsed", "ids")  
)
```

*Arguments:*

`comment` The comment to search for.

`retfmt` The format to use for the returned value. 'plain' will return the raw result from the server, as a character value. 'request' will return a `BiodbRequest` instance containing the request as it would have been sent. 'parsed' will return an XML object, containing the parsed result. 'ids' will return a character vector containing the IDs of the matching entries.

*Returns:* Depending on retfmt.

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

### See Also

[BiodbConn](#).

### Examples

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

# Get a connector:
conn <- mybiodb$getFactory()$createConn('expasy.enzyme')

# Get the first entry
e <- conn$getEntry('1.1.1.1')

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

---

ExpasyEnzymeEntry      *Expasy ENZYME database. entry class.*

---

### Description

Entry class for Expasy ENZYME database.

### Super classes

[biodb::BiodbEntry](#) -> [biodb::BiodbTxtEntry](#) -> ExpasyEnzymeEntry

### Methods

#### Public methods:

- [ExpasyEnzymeEntry\\$clone\(\)](#)

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

**See Also**

[BiodbTxtEntry](#).

**Examples**

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

# Get a connector that inherits from ExpasyEnzymeConn:
conn <- mybiodb$getFactory()$createConn('expasy.enzyme')

# Get the first entry
e <- conn$getEntry('1.1.1.1')

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

# Index

`biodb::BiodbConn`, 2  
`biodb::BiodbConnBase`, 2  
`biodb::BiodbEntry`, 4  
`biodb::BiodbTxtEntry`, 4  
`BiodbConn`, 4  
`biodbExpasy (biodbExpasy-package)`, 2  
`biodbExpasy-package`, 2  
`BiodbFactory`, 2  
`BiodbTxtEntry`, 5

`ExpasyEnzymeConn`, 2, 2  
`ExpasyEnzymeEntry`, 4