

# Package ‘biodbNci’

May 14, 2024

**Title** biodbNci, a library for connecting to biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database

**Version** 1.8.0

**Description** The biodbNci library is an extension of the biodb framework package. It provides access to biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database. It allows to retrieve entries by their accession number, and run specific web services.

**License** AGPL-3

**biocViews** Software, Infrastructure, DataImport

**VignetteBuilder** knitr

**Encoding** UTF-8

**Depends** R (>= 4.1)

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

**LinkingTo** Rcpp, testthat

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

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.2

**Collate** 'catch-routine-registration.R' 'NciCactusConn.R'  
'NciCactusEntry.R' 'RcppExports.R' 'package.R'

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

**git\_branch** RELEASE\_3\_19

**git\_last\_commit** 0db4525

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

**Repository** Bioconductor 3.19

**Date/Publication** 2024-05-14

**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

biodbNci-package . . . . .	2
NciCactusConn . . . . .	2
NciCactusEntry . . . . .	5

<b>Index</b>	<b>7</b>
--------------	----------

---

biodbNci-package	<i>biodbNci: biodbNci, a library for connecting to biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database</i>
------------------	--------------------------------------------------------------------------------------------------------------------------------------------------

---

### Description

The biodbNci library is an extension of the biodb framework package. It provides access to biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database. It allows to retrieve entries by their accession number, and run specific web services.

### Details

See vignette biodbNci:

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

### Author(s)

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

### See Also

[NciCactusConn](#).

---

NciCactusConn	<i>biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database. connector class.</i>
---------------	--------------------------------------------------------------------------------------------------------------------

---

### Description

biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database. connector class.

biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database. connector class.

## Details

Connector class for biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database.

This class implements a connector for accessing the NCI database, using CACTUS services. See <https://www.cancer.gov/> and <https://cactus.nci.nih.gov/>.

## Super classes

`biodb::BiodbConnBase` -> `biodb::BiodbConn` -> `NciCactusConn`

## Methods

### Public methods:

- `NciCactusConn$new()`
- `NciCactusConn$wsChemicalIdentifierResolver()`
- `NciCactusConn$conv()`
- `NciCactusConn$convCasToInchi()`
- `NciCactusConn$convCasToInchikey()`
- `NciCactusConn$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:*

```
NciCactusConn$new(...)
```

*Arguments:*

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

*Returns:* Nothing.

**Method** `wsChemicalIdentifierResolver()`: Calls Chemical Identifier Resolver web service. See [https://cactus.nci.nih.gov/chemical/structure\\_documentation](https://cactus.nci.nih.gov/chemical/structure_documentation) for details.

*Usage:*

```
NciCactusConn$wsChemicalIdentifierResolver(  
  structid,  
  repr,  
  xml = FALSE,  
  retfmt = c("plain", "parsed", "ids", "request")  
)
```

*Arguments:*

`structid` The submitted structure identifier.

`repr` The wanted representation.

`xml` A flag for choosing the format returned by the web service between plain text and XML.

`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 an XML 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** `conv()`: Calls `wsChemicalIdentifierResolver()` to convert a list of IDs into another representation.

*Usage:*

```
NciCactusConn$conv(ids, repr)
```

*Arguments:*

`ids` A character vector containing IDs.

`repr` The targeted representation.

*Returns:* A character vector, the same length as `ids`, containing the converted IDs. NA values will be set when conversion is not possible.

**Method** `convCasToInchi()`: Converts a list of CAS IDs into a list of InChI.

*Usage:*

```
NciCactusConn$convCasToInchi(cas)
```

*Arguments:*

`cas` A character vector containing CAS IDs.

*Returns:* A character vector, the same length as `ids`, containing InChI values or NA values where conversion was not possible.

**Method** `convCasToInchikey()`: Converts a list of CAS IDs into a list of InChI keys.

*Usage:*

```
NciCactusConn$convCasToInchikey(cas)
```

*Arguments:*

`cas` A character vector containing CAS IDs.

*Returns:* A character vector, the same length as `ids`, containing InChI Key values or NA values where conversion was not possible.

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

*Usage:*

```
NciCactusConn$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('nci.cactus')

# Use a database extract in order to avoid the downloading of the whole
# database.
dbExtract <- system.file("extdata", 'generated', "cactus_extract.txt.gz",
  package="biodbNci")
conn$setPropValSlot('urls', 'db.gz.url', dbExtract)

# Get an entry
e <- conn$getEntry('749674')

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

---

NciCactusEntry	<i>biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database. entry class.</i>
----------------	----------------------------------------------------------------------------------------------------------------

---

## Description

Entry class for biodbNci, a library for connecting to the National Cancer Institute (USA) CACTUS Database.

## Super classes

[biodb::BiodbEntry](#) -> [biodb::BiodbTxtEntry](#) -> [biodb::BiodbSdfEntry](#) -> NciCactusEntry

## Methods

### Public methods:

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

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

[BiodbSdfEntry](#).

**Examples**

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

# Get a connector that inherits from NciCactusConn:
conn <- mybiodb$getFactory()$createConn('nci.cactus')

# Use a database extract in order to avoid the downloading of the whole
# database.
dbExtract <- system.file("extdata", 'generated', "cactus_extract.txt.gz",
  package="biodbNci")
conn$setPropValSlot('urls', 'db.gz.url', dbExtract)

# Get an entry
e <- conn$getEntry('749674')

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

# Index

`biodb::BiodbConn`, 3  
`biodb::BiodbConnBase`, 3  
`biodb::BiodbEntry`, 5  
`biodb::BiodbSdfEntry`, 5  
`biodb::BiodbTxtEntry`, 5  
`BiodbConn`, 4  
`biodbNci` (`biodbNci-package`), 2  
`biodbNci-package`, 2  
`BiodbSdfEntry`, 5  
  
`NciCactusConn`, 2, 2  
`NciCactusEntry`, 5