

# Package ‘miRNANameConverter’

April 16, 2024

**Type** Package

**Title** Convert miRNA Names to Different miRBase Versions

**Version** 1.30.0

**Description** Translating mature miRNA names to different miRBase versions, sequence retrieval, checking names for validity and detecting miRBase version of a given set of names (data from <http://www.mirbase.org/>).

**License** Artistic-2.0

**Imports** DBI, AnnotationDbi, reshape2

**Depends** miRBaseVersions.db

**biocViews** Preprocessing, miRNA

**LazyData** TRUE

**Suggests** methods, testthat, knitr, rmarkdown

**VignetteBuilder** knitr

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Stefan Haunsberger [aut, cre]

**Maintainer** Stefan J. Haunsberger <[stefan.haunsberger@gmail.com](mailto:stefan.haunsberger@gmail.com)>

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

**git\_branch** RELEASE\_3\_18

**git\_last\_commit** 40e10f7

**git\_last\_commit\_date** 2023-10-24

**Repository** Bioconductor 3.18

**Date/Publication** 2024-04-15

## R topics documented:

<code>assessMiRNASwappingMIMAT</code> . . . . .	2
<code>assessVersion</code> . . . . .	3

checkMiRNAName . . . . .	4
currentVersion . . . . .	5
currentVersion<- . . . . .	6
example.miRNAs . . . . .	6
miRNANameConverter . . . . .	7
MiRNANameConverter . . . . .	8
MiRNANameConverter,ANY-method . . . . .	8
nOrganisms . . . . .	9
nOrganisms<- . . . . .	10
nTotalEntries . . . . .	10
nTotalEntries<- . . . . .	11
saveResults . . . . .	12
show,MiRNANameConverter-method . . . . .	13
translateMiRNAName . . . . .	14
validOrganisms . . . . .	15
validOrganisms<- . . . . .	16
validVersions . . . . .	17
validVersions<- . . . . .	17

## Index 19

---

### assessMiRNASwappingMIMAT

*Check if given miRNA names can be assigned to unique MIMAT accessions among all versions*

---

#### Description

This function checks if the names from a given set of mature miRNAs have a unique MIMAT ID. Check if given miRNA names can be assigned to unique MIMAT accessions among all versions

This function checks if the names from a given set of mature miRNAs have a unique MIMAT ID.

#### Usage

```
assessMiRNASwappingMIMAT(this, miRNAs, verbose = FALSE)
```

#### Arguments

this	Object of class 'MiRNANameConverter'
miRNAs	A character vector of miRNA names
verbose	A boolean to either show more (TRUE) or less information (FALSE)

#### Details

Although the majority of miRNA names can be assigned to a unique MIMAT ID (accession) some miRNAs changed MIMAT ID in different versions. This function takes the input miRNA names and checks each one of them if they have a unique MIMAT ID over all versions. If a miRNA changes MIMAT ID in a version it will be comprised in the return vector.

**Value**

A character vector containing miRNA names that do not have a unique MIMAT ID

**Author(s)**

Stefan Haunsberger

assessVersion                      *Assess miRBase version*

**Description**

This function detects the most likely miRBase version of a given miRNA set. Assess miRBase version

This function detects the most likely miRBase version of a given miRNA set.

**Usage**

```
assessVersion(this, miRNAs, verbose = FALSE)

## S4 method for signature 'MiRNANameConverter'
assessVersion(this, miRNAs, verbose = FALSE)
```

**Arguments**

this                      Object of class 'MiRNANameConverter'  
 miRNAs                    A character vector of miRNA names  
 verbose                   A boolean to either show more (TRUE) or less information (FALSE)

**Details**

This function takes a set of miRNA names and detects the most likely miRBase version of this given set of 'miRNAs'. First all miRNAs will be checked for validity (if they are actual miRNA names checkMiRNAName and the set that passes the check will be further processed.

**Value**

A data frame with two columns: version and frequency (decreasing order by frequency, version) + version: miRBase version + frequency: the number of valid miRNAs that could be assigned to the version respectively

**Methods (by class)**

- MiRNANameConverter: Method for assessing the most likely miRBase version that a given set of miRNA names is from.

**Author(s)**

Stefan Haunsberger

**Examples**

```
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'
assessVersion(nc, miRNAs = c("hsa-miR-140", "hsa-miR-125a"))
```

---

checkMiRNAName	<i>Check miRNA names for validity</i>
----------------	---------------------------------------

---

**Description**

This function checks for a given set of mature 'miRNAs' (names) Check miRNA names for validity

This function checks for a given set of mature 'miRNAs' (names) if the names are listed in any miRBase version respectively.

**Usage**

```
checkMiRNAName(this, miRNAs, verbose = FALSE)

## S4 method for signature 'MiRNANameConverter'
checkMiRNAName(this, miRNAs, verbose = FALSE)
```

**Arguments**

this	Object of class 'MiRNANameConverter'
miRNAs	A character vector of miRNA names
verbose	A boolean to either show more (TRUE) or less information (FALSE)

**Details**

This function takes the input miRNA names and checks each one of them for validity. The check is done by taking each miRNA and searches for an existing entry in the miRBase database among all versions. miRNAs that are listed in any version will be comprised in the return vector respectively. If no valid miRNA was detected, a character(0) will be returned.

**Value**

A character vector containing a set of valid miRNA names

**Methods (by class)**

- MiRNANameConverter: Method for checking for valid miRNA names

**Author(s)**

Stefan Haunsberger

**Examples**

```
nc = MiRNANameConverter() # Instance of class 'MiRNANameConverter'  
# Test with correct inputs  
checkMiRNAName(nc, miRNAs = c("hsa-miR-29a", "hsa-miR-642"))
```

---

currentVersion	<i>Get current version</i>
----------------	----------------------------

---

**Description**

This function returns the highest miRBase version that is provided by the package.

**Usage**

```
currentVersion(this)  
  
## S4 method for signature 'MiRNANameConverter'  
currentVersion(this)
```

**Arguments**

this            Object of class MiRNANameConverter

**Details**

The maximum miRBase version of the package is evaluated and set in the object initialization.

**Value**

A numeric value

**Methods (by class)**

- MiRNANameConverter: Retrieve highest supported miRBase version

**Author(s)**

Stefan Haunsberger

**Examples**

```
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'  
currentVersion(nc);
```

currentVersion<-        *Set current version*

---

**Description**

Set the highest version that is supported by the package.

**Usage**

```
currentVersion(this) <- value
```

**Arguments**

this	Object of class ”
value	A numeric value

**Details**

The value for the highest version is a static variable. It is initialized in the initialization method when an instance of a MiRNANameConverter class is created.

**Value**

Object of class ”

**Author(s)**

Stefan Haunsberger

---

example.miRNAs        *miRNA names.*

---

**Description**

Sample names including miRNA names, non-miRNA names and other. It also includes duplicates.

**Usage**

```
example.miRNAs
```

**Format**

A character vector containing names.

---

miRNAmeConverter      *MiRNANameConverter* constructor

---

## Description

This function returns an instance of a `MiRNANameConverter` class. Handling mature miRNA names from different miRBase versions This package contains algorithms for dealing with mature miRNA names from different miRBase release versions. The functions are provided in form of methods as part of the `MiRNANameConverter`-class. The data of all the miRBase release versions is stored in the `miRBaseVersions.db` annotation package. The *MiRNAmeConverter* package contains one class that has two categories of functions: getters-functions and algorithms.

## Classes

The `MiRNANameConverter`

## Getter functions

The getter functions provide access to the slots of the class.

## Algorithms

There are three algorithms for dealing with miRNA names from different miRBase releases, the [assessVersion](#), [checkMiRNAName](#) and [translateMiRNAName](#).

[translateMiRNAName](#) The algorithm coded in this function can translate given miRNA names to different miRBase release versions.

[checkMiRNAName](#) This function is used to check if a given miRNA name is listed in the current miRBase release.

[assessVersion](#) The *assessVersion*-function is useful when one wants to assess the miRBase version of a given set of mature miRNA names.

## Author(s)

Stefan Haunsberger <stefanhaunsberger@rcsi.ie>

## See Also

*miRBaseVersions.db* for more information about the database holding all major miRBase release versions)

## Examples

```
# Translate a mature miRNA name to miRBase version 21.0
nc = MiRNANameConverter(); # Object instantiation
translateMiRNAName(nc, "hsa-miR-29a", version = 21.0)
```

MiRNANameConverter     *Instantiate from MiRNANameConverter class*

---

**Description**

This function returns back an instance of a *MiRNANameConverter* object.

**Usage**

```
MiRNANameConverter(...)
```

**Arguments**

...                    any optional arguments

**Slots**

.dbconn    Database connection  
.currentVersion    Current miRBase version  
.validVersions    Valid/Supported miRBase versions  
.nOrganisms    Number of different organisms supported  
.nTotalEntries    Total number of mature miRNA names among all provided miRBase release versions in the *miRBaseVersions.db* package.  
.validOrganisms    Valid organisms

**Author(s)**

Stefan Haunsberger

---

MiRNANameConverter, ANY-method

*MiRNANameConverter constructor*

---

**Description**

This function returns an instance of a *MiRNANameConverter* class.

**Usage**

```
## S4 method for signature 'ANY'  
MiRNANameConverter()
```

**Details**

This function initializes an object of the class *MiRNANameConverter*. It is a wrapper for `new()`.



**Value**

an object of class 'MiRNANameConverter'

**Author(s)**

Stefan Haunsberger

**See Also**

[new](#)

**Examples**

```
nc = MiRNANameConverter() # Instance of class 'MiRNANameConverter'
```

---

nOrganisms

*Get number of organisms*

---

**Description**

This function returns the number of different organisms that are provided by the package.

**Usage**

```
nOrganisms(this)

## S4 method for signature 'MiRNANameConverter'
nOrganisms(this)
```

**Arguments**

this            Object of class MiRNANameConverter

**Details**

The number of different organisms is evaluated and set in the object initialization.

**Value**

A numeric value

**Methods (by class)**

- MiRNANameConverter: Retrieve number of organisms

**Author(s)**

Stefan Haunsberger

**Examples**

```
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'  
nOrganisms(nc);
```

---

```
nOrganisms<-          Set number of organisms
```

---

**Description**

This function sets the number of different organisms that are provided by the package.

**Usage**

```
nOrganisms(this) <- value
```

**Arguments**

this	Object of class MiRNANameConverter
value	An integer value

**Details**

The number of different organisms is evaluated and set in the object initialization.

**Value**

A MiRNANameConverter object

**Author(s)**

Stefan Haunsberger

---

```
nTotalEntries          Get total number database entries
```

---

**Description**

This function returns the total number of entries contained in the mimat table. The number is the sum of the entries of all miRBase versions provided by the package.

**Usage**

```
nTotalEntries(this)
```

```
## S4 method for signature 'MiRNANameConverter'  
nTotalEntries(this)
```

**Arguments**

    this            Object of class MiRNANameConverter

**Details**

    The total number is evaluated and set in the object initialization.

**Value**

    A numeric value

**Methods (by class)**

- MiRNANameConverter: Retrieve total number of miRNA entries

**Author(s)**

    Stefan Haunsberger

**Examples**

```
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'  
nTotalEntries(nc);
```

---

*nTotalEntries*<-            *Set total number database entries*

---

**Description**

    This function sets the total number of entries contained in the mimat table. The number is the sum of the entries of all miRBase versions provided by the package.

**Usage**

```
nTotalEntries(this) <- value
```

**Arguments**

    this            Object of class MiRNANameConverter  
    value           An integer value

**Details**

    The total number is evaluated and set in the object initialization.

**Value**

    A MiRNANameConverter object

**Author(s)**

Stefan Haunsberger

---

saveResults	<i>Save miRNA translation results</i>
-------------	---------------------------------------

---

**Description**

This function saves the data frame returned from `translateMiRNAName` inclusive the attribute 'description'. Save miRNA translation results

This function saves the data frame returned from `translateMiRNAName` inclusive the attribute 'description'.

**Usage**

```
saveResults(this, df, outputFilename, outputPath, sep = "\t",
            quote = FALSE, verbose = FALSE, ...)
```

```
## S4 method for signature 'MiRNANameConverter,data.frame'
saveResults(this, df, outputFilename,
            outputPath, sep = "\t", quote = FALSE, verbose = FALSE, ...)
```

**Arguments**

<code>this</code>	Object of class 'MiRNANameConverter'
<code>df</code>	A data.frame with translated results
<code>outputFilename</code>	A filename for the output file, such as 'filename.txt'
<code>outputPath</code>	A file path (character string) to the target directory
<code>sep</code>	Separator
<code>quote</code>	If all data values shall be surrounded by ('')
<code>verbose</code>	Boolean to either show more (TRUE) or less information (FALSE)
<code>...</code>	Arguments that can be passed on to <code>write.table</code>

**Details**

This function saves a data frame that has been returned by `translateMiRNAName`. The attribute 'description' of the data frame will be stored as well.

**Methods (by class)**

- `this = MiRNANameConverter, df = data.frame`: Method for saving translation results

**Author(s)**

Stefan Haunsberger

**See Also**

[write.table](#) for additional parameter values for the '...' argument, [attr](#) for how to retrieve attributes

**Examples**

```
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'
res = translateMiRNAName(nc, miRNAs = c("hsa-miR-140", "hsa-miR-125a"),
                        versions = c(15, 16, 20, 21))

# Save translation results
saveResults(nc, res)
```

---

show, MiRNANameConverter-method  
*Show-method*

---

**Description**

This function prints object specific information Show-method

This function prints object specific information

**Usage**

```
## S4 method for signature 'MiRNANameConverter'
show(object)
```

**Arguments**

object            Object of class MiRNANameConverter

**Details**

This function prints some information to the console.

**Author(s)**

Stefan Haunsberger

**See Also**

[show](#)

---

translateMiRNAName	<i>Translate miRNA name</i>
--------------------	-----------------------------

---

### Description

This function translates input miRNA names to different miRBase versions. Translate miRNA name

This function translates input miRNA names to different miRBase versions.

### Usage

```
translateMiRNAName(this, miRNAs, versions, sequenceFormat = 1,
  verbose = FALSE)
```

```
## S4 method for signature 'MiRNANameConverter,character'
translateMiRNAName(this, miRNAs,
  versions, sequenceFormat = 1, verbose = FALSE)
```

### Arguments

this	Object of class 'MiRNANameConverter'
miRNAs	A character vector of miRNA names
versions	An integer or numerical vector containing the target versions
sequenceFormat	Integer value indicating the return format for the data frame containing sequence information 1=only sequences, 2=miRNA name and sequence
verbose	A boolean to either show more (TRUE) or less information (FALSE)

### Details

The translation and sequence retrieval are done in 5 main steps: 1) Only take miRNA names that do not swap MIMAT IDs among versions ([assessMiRNASwappingMIMAT](#)) 2) Check, if the miRNA names are valid names ([checkMiRNAName](#)) 3) Receive unique MIMAT IDs for each valid miRNA - If there are miRNAs that have basically the same name, only use miRNA names from the highest version 4) Check if the found MIMAT IDs are still listed in the current miRBase version - If not, neglect it because then it is not considered to be a miRNA anymore 5) Receive names from desired versions

### Value

A (n x m) data frame where n is the number of valid miRNAs and m the number of columns (minimum 3 columns, MIMAT-ID (accession), input miRNA name, current version) In addition an attribute 'description' is added to the data frame where to each miRNA some notes are added (for example why a certain miRNA is not in the output). Sequence information is attached as the attribute 'sequence'.

**Methods (by class)**

- `this = MiRNANameConverter, miRNAs = character`: Method for translating miRNA name(s) to different miRBase versions

**Author(s)**

Stefan Haunsberger

**See Also**

[attr](#) for attributes

**Examples**

```
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'  
res = translateMiRNAName(nc, miRNAs = c("hsa-miR-140", "hsa-miR-125a"),  
                          versions = c(15, 16, 20, 21))  
  
res  
attributes(res)
```

---

validOrganisms

*Get valid organisms*

---

**Description**

This function returns all organisms where mature miRNA names are available in any of the provided miRBase versions.

**Usage**

```
validOrganisms(this)
```

```
## S4 method for signature 'MiRNANameConverter'  
validOrganisms(this)
```

**Arguments**

`this`            Object of class `MiRNANameConverter`

**Details**

The valid organisms are evaluated and set in the object initialization.

**Value**

A numeric value

**Methods (by class)**

- `MiRNANameConverter`: Retrieve all supported organisms

**Author(s)**

Stefan Haunsberger

**Examples**

```
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'  
nOrganisms(nc);
```

---

```
validOrganisms<-      Set valid organisms
```

---

**Description**

This function sets all organisms where mature miRNA names are available in any of the provided miRBase versions.

**Usage**

```
validOrganisms(this) <- value
```

**Arguments**

<code>this</code>	Object of class <code>MiRNANameConverter</code>
<code>value</code>	A character vector

**Details**

The valid organisms are evaluated and set in the object initialization.

**Value**

A `MiRNANameConverter` object

**Author(s)**

Stefan Haunsberger



---

validVersions	<i>Get valid versions</i>
---------------	---------------------------

---

**Description**

This function returns all valid miRBase versions provided by the package.

**Usage**

```
validVersions(this)

## S4 method for signature 'MiRNANameConverter'
validVersions(this)
```

**Arguments**

this            Object of class MiRNANameConverter

**Value**

A numeric vector

**Methods (by class)**

- MiRNANameConverter: Retrieve supported miRBase versions

**Author(s)**

Stefan Haunsberger

**Examples**

```
validVersions
nc = MiRNANameConverter(); # Instance of class 'MiRNANameConverter'
validVersions(nc);
```

---

validVersions<-	<i>Set valid versions</i>
-----------------	---------------------------

---

**Description**

Set version values that are supported by the package.

**Usage**

```
validVersions(this) <- value
```

**Arguments**

<code>this</code>	Object of class <code>MiRNANameConverter</code>
<code>value</code>	A vector of numeric values

**Details**

The value for the highest versions is a static variable. It is initialized in the initialization method when an instance of a `MiRNANameConverter` class is created.

**Value**

A `MiRNANameConverter` object

**Author(s)**

Stefan Haunsberger

# Index

- \* **datasets**
  - example.miRNAs, 6
- assessMiRNASwappingMIMAT, 2, 14
- assessVersion, 3, 7
- assessVersion, MiRNANameConverter-method (assessVersion), 3
- attr, 13, 15
- checkMiRNAName, 4, 7, 14
- checkMiRNAName, MiRNANameConverter-method (checkMiRNAName), 4
- currentVersion, 5
- currentVersion, MiRNANameConverter-method (currentVersion), 5
- currentVersion<-, 6
- example.miRNAs, 6
- miRNANameConverter, 7
- miRNANameConverter-package (miRNANameConverter), 7
- MiRNANameConverter, 8
- MiRNANameConverter, ANY-method, 8
- new, 9
- nOrganisms, 9
- nOrganisms, MiRNANameConverter-method (nOrganisms), 9
- nOrganisms<-, 10
- nTotalEntries, 10
- nTotalEntries, MiRNANameConverter-method (nTotalEntries), 10
- nTotalEntries<-, 11
- saveResults, 12
- saveResults, MiRNANameConverter, data.frame-method (saveResults), 12
- show, 13
- show, MiRNANameConverter-method, 13
- translateMiRNAName, 7, 14
- translateMiRNAName, MiRNANameConverter, character-method (translateMiRNAName), 14
- validOrganisms, 15
- validOrganisms, MiRNANameConverter-method (validOrganisms), 15
- validOrganisms<-, 16
- validVersions, 17
- validVersions, MiRNANameConverter-method (validVersions), 17
- validVersions<-, 17
- write.table, 13