

# Package ‘BiocBaseUtils’

February 6, 2023

**Title** General utility functions for developing Bioconductor packages

**Version** 1.0.0

**Description** The package provides utility functions related to package development. These include functions that replace slots, and selectors for show methods. It aims to coalesce the various helper functions often re-used throughout the Bioconductor ecosystem.

**Imports** methods, utils

**Depends** R (>= 4.2.0)

**Suggests** knitr, rmarkdown, BiocStyle, tinytest

**License** Artistic-2.0

**Encoding** UTF-8

**biocViews** Software, Infrastructure

**BugReports** <https://www.github.com/Bioconductor/BiocBaseUtils/issues>

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.1

**VignetteBuilder** knitr

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

**git\_branch** RELEASE\_3\_16

**git\_last\_commit** c6c2cde

**git\_last\_commit\_date** 2022-11-01

**Date/Publication** 2023-02-06

**Author** Marcel Ramos [aut, cre] (<<https://orcid.org/0000-0002-3242-0582>>),  
Martin Morgan [ctb],  
Hervé Pagès [ctb]

**Maintainer** Marcel Ramos <[marcel.ramos@roswellpark.org](mailto:marcel.ramos@roswellpark.org)>

## R topics documented:

BiocBaseUtils-package . . . . .	2
Assertions . . . . .	2
selectSome . . . . .	4
setSlots . . . . .	5
<b>Index</b>	<b>7</b>

---

BiocBaseUtils-package *BiocBaseUtils: Utility and Internal functions for Bioconductor packages*

---

### Description

BiocBaseUtils is a package aimed at helping the typical Bioconductor developer formalize often written functions that can be seen scattered throughout the Bioconductor ecosystem. Some of these functions include the ability to replace slots in an object. Other functions work to create a nice show method output by selecting some observations.

### Author(s)

**Maintainer:** Marcel Ramos <marcel.ramos@roswellpark.org> ([ORCID](#))

Other contributors:

- Martin Morgan <martin.morgan@roswellpark.org> [contributor]
- Hervé Pagès <hpages.on.github@gmail.com> [contributor]

### See Also

Useful links:

- Report bugs at <https://www.github.com/Bioconductor/BiocBaseUtils/issues>

---

Assertions *Suite of helper functions to test for types*

---

### Description

These are a group of helper functions that allow the developer to easily check for common data types in Bioconductor. These include logical, character, and numeric (& integer).

**Usage**

```
isTRUEorFALSE(x, na.ok = FALSE)

isScalarCharacter(x, na.ok = FALSE, zchar = FALSE)

isScalarInteger(x, na.ok = FALSE)

isScalarNumber(x, na.ok = FALSE, infinite.ok = FALSE)

isCharacter(x, na.ok = FALSE, zchar = FALSE)

isZeroOneCharacter(x, na.ok = FALSE, zchar = FALSE)
```

**Arguments**

x	The input vector whose type is to be checked
na.ok	logical(1L) Whether it is acceptable to consider NA type inputs (default: FALSE).
zchar	logical(1L) Whether it is acceptable to consider 'zero' characters as defined by nchar, e.g., nchar("") (default: FALSE).
infinite.ok	logical(1L) Whether it is acceptable to consider infinite values as identified by is.finite (default: FALSE).

**Details**

Some functions such as `isScalarCharacter` allow exceptions to the type checks via the `na.ok` and `zchar` arguments. Others, for example `isScalarNumber` can permit `Inf` with the `infinite.ok` argument.

**Value**

Either TRUE or FALSE

**Functions**

- `isTRUEorFALSE()`: Is the input a single logical vector?
- `isScalarCharacter()`: Is the input a single character vector?
- `isScalarInteger()`: Is the input a single integer vector?
- `isScalarNumber()`: Is the input a single numeric vector?
- `isCharacter()`: Is the input a character vector?
- `isZeroOneCharacter()`: Is the input a character vector of zero or one length?

**Author(s)**

M. Morgan, H. Pagès

**Examples**

```

isTRUEorFALSE(TRUE)
isTRUEorFALSE(FALSE)
isTRUEorFALSE(NA, na.ok = TRUE)

isScalarCharacter(LETTERS)
isScalarCharacter("L")
isCharacter(LETTERS)
isCharacter(NA_character_, na.ok = TRUE)
isZeroOneCharacter("")
isZeroOneCharacter("", zchar = TRUE)

isScalarInteger(1L)
isScalarInteger(1)

isScalarNumber(1)
isScalarNumber(1:2)

```

---

selectSome

*Select and return only some entries from a vector*


---

**Description**

selectSome works well in show methods. It abbreviates a vector input depending on the maxToShow argument.

**Usage**

```

selectSome(
  obj,
  maxToShow = 5,
  ellipsis = "...",
  ellipsisPos = c("middle", "end", "start"),
  quote = FALSE
)

```

**Arguments**

obj	character() A vector to be abbreviated for display purposes
maxToShow	numeric(1) The maximum number of values to show in the output (default: 5)
ellipsis	character(1) The symbol used to abbreviate values in the vector (default: "...")
ellipsisPos	character(1) The location for the ellipsis in the output, by default in the "middle" but can be moved to either the "end" or the "start".
quote	logical(1) Whether or not to add a single quote around the obj input. This only works for character type inputs.

**Value**

An abbreviated output of obj

**Author(s)**

M. Morgan, H. Pagès

**Examples**

```
letters
```

```
selectSome(letters)
```

---

setSlots

*Convenience function to set slot values*

---

**Description**

Given the current object, the function `setSlots` will take name-value pair inputs either as named arguments or a list and replace the values of the specified slots. This is a convenient function for updating slots in an S4 class object.

**Usage**

```
setSlots(object, ..., check = TRUE)
```

**Arguments**

object	An S4 object with slots to replace
...	Slot name and value pairs either as named arguments or a named list, e.g., <code>slotName = value</code> .
check	logical(1L) Whether to run <code>validObject</code> after the slot replacement

**Value**

The object input with updated slot data

**Author(s)**

H. Pagès

**Examples**

```
setClass("A", representation = representation(slotA = "character"))  
  
aclass <- new("A", slotA = "A")  
  
setSlots(aclass, slotA = "B")
```

# Index

Assertions, [2](#)

BiocBaseUtils-package, [2](#)

isCharacter (Assertions), [2](#)

isScalarCharacter (Assertions), [2](#)

isScalarInteger (Assertions), [2](#)

isScalarNumber (Assertions), [2](#)

isTRUEorFALSE (Assertions), [2](#)

isZeroOneCharacter (Assertions), [2](#)

replaceSlots (setSlots), [5](#)

selectSome, [4](#)

setSlots, [5](#)