

# Package ‘kimisc’

October 13, 2022

**Encoding** UTF-8

**Type** Package

**Title** Kirill's Miscellaneous Functions

**Version** 0.4

**Date** 2017-12-17

**Description** A collection of useful functions not found anywhere else, mainly for programming: Pretty intervals, generalized lagged differences, checking containment in an interval, and an alternative interface to assign().

**License** GPL-3

**Imports** memoise, plyr, pryr

**Suggests** testthat (>= 0.10.0)

**Enhances** knitr

**URL** <http://kr1mlr.github.io/kimisc>

**URLNote** <https://github.com/kr1mlr/kimisc>

**BugReports** <https://github.com/kr1mlr/kimisc/issues>

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Kirill Müller [aut, cre]

**Maintainer** Kirill Müller <kr1mlr+r@mailbox.org>

**Repository** CRAN

**Date/Publication** 2017-12-18 16:13:51 UTC

## R topics documented:

kimisc-package . . . . .	2
coalesce.na-deprecated . . . . .	3
cut_format . . . . .	3
df_to_list-deprecated . . . . .	4

export . . . . .	5
export.list . . . . .	6
gdiff . . . . .	7
hms.to.seconds-deprecated . . . . .	8
in.interval.lo . . . . .	9
in.interval.ro . . . . .	9
kimisc-deprecated . . . . .	10
list_to_df-deprecated . . . . .	10
nc-deprecated . . . . .	11
nin.interval.lo . . . . .	12
nin.interval.ro . . . . .	13
nlist-deprecated . . . . .	13
ofactor-deprecated . . . . .	14
sample.rows-deprecated . . . . .	15
seconds.to.hms-deprecated . . . . .	16
thisfile-deprecated . . . . .	17
tll-deprecated . . . . .	18
vswitch-deprecated . . . . .	19
<b>Index</b>	<b>20</b>

---

kimisc-package	<i>kimisc: Kirill's Miscellaneous Functions</i>
----------------	---

---

## Description

A collection of useful functions not found anywhere else, mainly for programming: Pretty intervals, generalized lagged differences, checking containment in an interval, and an alternative interface to `assign()`.

## Author(s)

Kirill Müller

## See Also

Useful links:

- <http://kr1mlr.github.io/kimisc>
- Report bugs at <https://github.com/kr1mlr/kimisc/issues>

---

`coalesce.na-deprecated`*Replaces NA values*

---

**Description**

This (vectorized) function returns the first non-NA argument, similar to the SQL function COALESCE. If a vector or matrix is passed as first argument, the remaining arguments are recycled to generate a vector/matrix of the same dimension, and coalescing is done element by element.

**Usage**

```
coalesce.na(x, ...)
```

**Arguments**

<code>x</code>	The first value to coalesce.
<code>...</code>	Other values to coalesce.

**Value**

A vector of the same length as `x`.

**See Also**

Other deprecated functions: [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

**Examples**

```
coalesce.na(NA, -1)
coalesce.na(5, 3)
coalesce.na(c(1,NA,NA), c(NA,2))
coalesce.na(matrix(c(NA, 1:3), nrow=2))
coalesce.na(NA)
```

---

`cut_format`*Convert Numeric to Factor, with custom formatting*

---

**Description**

This is an enhanced version of `base::cut()` that allows a custom formatting to be applied to the values.

**Usage**

```
cut_format(x, breaks, include.lowest = FALSE, right = TRUE,
  ordered_result = FALSE, ..., format_fun = format, sep = ", ",
  paren = c("(", "[", ")", "]"))
```

**Arguments**

x	a numeric vector which is to be converted to a factor by cutting.
breaks	[numeric] A vector of two or more unique cut points
include.lowest	logical, indicating if an 'x[i]' equal to the lowest (or highest, for right = FALSE) 'breaks' value should be included.
right	logical, indicating if the intervals should be closed on the right (and open on the left) or vice versa.
ordered_result	logical: should the result be an ordered factor?
...	Passed to cut()
format_fun	[function(x): character] A vectorized function that performs the desired formatting. Default: <code>base::format()</code>
sep	[character(1)] The separator between lower and upper end of the interval. Default: ", "
paren	[character(4)] Opening and closing parentheses in two variants. Default: <code>c("(", "[", ")", "]")</code>

**See Also**

<http://stackoverflow.com/q/14456371/946850>

**Examples**

```
cut_format(runif(10), seq(0, 1, by = 0.25), format_fun = function(x) paste(x * 100, "%"))
cut_format(runif(10), seq(0, 1, by = 0.25), paren = c("<", "{", ">", "}"))
```

---

df\_to\_list-deprecated *Converts a name-value data frame to a named list*

---

**Description**

This function converts a data frame back to a list. It is the reverse operation to [list\\_to\\_df](#).

**Usage**

```
df_to_list(df_for_list)
```

**Arguments**

`df_for_list`      The data frame to be converted to a list

**Details**

In a data frame with more than two columns, heuristics are applied to detect the name and value column.

**See Also**

Other deprecated functions: [coalesce.na-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

---

export

*Exports to an environment*

---

**Description**

This function is a wrapper around [export.list\(\)](#) that exports variables by their name to another environment.

**Usage**

```
export(..., target.env = .GlobalEnv)
```

**Arguments**

`...`              variables to be exported.  
`target.env`        The target environment. Use the global environment by default.

**Value**

Invisible NULL.

**Author(s)**

Roland

**References**

<http://stackoverflow.com/a/17484932/946850>

**See Also**

[export.list](#), [assign](#)

## Examples

```
local({
  newly.created.var <- 5
  export(newly.created.var)
})
newly.created.var
rm(newly.created.var)
```

---

export.list

*Exports to an environment*

---

## Description

This function is a wrapper around `assign()` that exports the contents of a named list to an environment. The variable names in the target environment are constructed from the names of the list items or taken from a separate argument.

## Usage

```
export.list(arg.list, arg.names = names(arg.list), target.env = .GlobalEnv)
```

## Arguments

<code>arg.list</code>	list of objects, possibly named.
<code>arg.names</code>	names to use for the items in the target environment. Use the names of <code>arg.list</code> by default.
<code>target.env</code>	The target environment. Use the global environment by default.

## Value

Invisible NULL.

## Author(s)

Roland

## References

<http://stackoverflow.com/a/17484932/946850>

## See Also

[export](#), [assign](#)

## Examples

```
export.list(list(newly.created.var=5))
newly.created.var
rm(newly.created.var)
```

---

gdiff	<i>Generalized lagged differences</i>
-------	---------------------------------------

---

### Description

Returns suitably lagged and iterated differences using arbitrary difference functions.

### Usage

```
gdiff(x, lag = 1L, differences = 1L, FUN = `-`, ...)
```

### Arguments

x	a numeric vector or matrix containing the values to be differenced.
lag	an integer indicating which lag to use.
differences	an integer indicating the order of the difference.
FUN	A distance function that accepts two parameters
...	further arguments to be passed to or from methods.

### Value

If x is a vector of length n and differences = 1, then the computed result is equal to the successive differences FUN(x[(1+lag):n], x[1:(n-lag)]).

If difference is larger than one this algorithm is applied recursively to x. Note that the returned value is a vector which is shorter than x.

If x is a matrix then the difference operations are carried out on each column separately.

### See Also

[base::diff\(\)](#)

### Examples

```
gdiff(1:4)
gdiff(1:4, FUN = `/'`)
```

hms.to.seconds-deprecated

*Converts a time value given in H:M:S format to the number of seconds since midnight*

---

### Description

This function is very similar to `strptime` with the `%X` conversion specification. Anything with three numbers between two colons is interpreted as a time, no consistency check is performed on the actual hour, minute and second values. Thus, strings like `25:15:00` and `23:78:101` also will be converted. Incorrectly formatted strings are converted to `NA` with a warning.

### Usage

```
hms.to.seconds(x)
```

### Arguments

`x` A (vector of) strings in H:M:S format.

### Value

A (vector of) integer values of the same length as `x`.

### See Also

[base::strptime\(\)](#)

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

### Examples

```
hms.to.seconds(c("00:00:01", "00:01:00", "01:00:00"))
hms.to.seconds(c("25:15:00", "23:78:101"))
hms.to.seconds("invalid")
```



---

in.interval.lo	<i>Checks if values are contained in an interval (open on the left)</i>
----------------	---

---

**Description**

This function checks if the values in the `x` parameter are contained in the interval  $(lo, hi]$ . NA values are treated as "not in the interval".

**Usage**

```
in.interval.lo(x, lo, hi)
```

**Arguments**

<code>x</code>	A vector of values. (Lists will be coerced to a numeric vector.)
<code>lo</code>	Left end of the interval.
<code>hi</code>	Right end of the interval.

**Value**

A boolean vector of the same length as `x`.

**See Also**

[in.interval.ro](#), [nin.interval.lo](#), [nin.interval.ro](#)

**Examples**

```
in.interval.lo(c(-1, 0, 1, 2), 0, 1)
in.interval.lo(NA, 1, 3)
```

---

in.interval.ro	<i>Checks if values are contained in an interval (open on the right)</i>
----------------	--

---

**Description**

This function checks if the values in the `x` parameter are contained in the interval  $[lo, hi)$ . NA values are treated as "not in the interval".

**Usage**

```
in.interval.ro(x, lo, hi)
```

**Arguments**

x	A vector of values. (Lists will be coerced to a numeric vector.)
lo	Left end of the interval.
hi	Right end of the interval.

**Value**

A boolean vector of the same length as x.

**See Also**

[in.interval.lo](#), [nin.interval.lo](#), [nin.interval.ro](#)

**Examples**

```
in.interval.ro(c(-1, 0, 1, 2), 0, 1)
in.interval.ro(NA, 1, 3)
```

---

kimisc-deprecated      *Deprecated functions*

---

**Description**

The "See also" section contains the deprecated functions in this package.

**See Also**

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

---

list\_to\_df-deprecated      *Converts a list to a name-value data frame*

---

**Description**

This function coerces its input to a list and returns a data frame with as many rows as there are list items in the input, and two columns (one for the names, one for the values). If the list is not named, the natural sequence will be used as item names.

**Usage**

```
list_to_df(list_for_df)
```

**Arguments**

`list_for_df`      The object to be converted to a data frame

**See Also**

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

---

nc-deprecated

*Smart named vector*

---

**Description**

This function is a wrapper around `c()` that assigns names to unnamed arguments based on the unevaluated expression used in the call.

**Usage**

```
nc(...)
```

**Arguments**

`...`      Vector elements, possibly named

**Value**

A named vector.

**Author(s)**

Hadley Wickham

**References**

<http://stackoverflow.com/a/5043280/946850>, <http://tolstoy.newcastle.edu.au/R/e9/help/10/03/8392.html>

**See Also**

[c](#), [nlist](#)

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

**Examples**

```
a <- 1; b <- 2; c <- 3
nc(a, b, d=c)
nc(mean(c(a, b, c)))
```

---

nin.interval.lo	<i>Checks if values are outside of an interval (open on the left)</i>
-----------------	---

---

**Description**

This function checks if the values in the `x` parameter are contained in the interval  $(lo, hi]$ . NA values are treated as "not in the interval".

**Usage**

```
nin.interval.lo(x, lo, hi)
```

**Arguments**

<code>x</code>	A vector of values. (Lists will be coerced to a numeric vector.)
<code>lo</code>	Left end of the interval.
<code>hi</code>	Right end of the interval.

**Value**

A boolean vector of the same length as `x`.

**See Also**

[in.interval.lo](#), [in.interval.ro](#), [nin.interval.ro](#)

**Examples**

```
nin.interval.lo(c(-1, 0, 1, 2), 0, 1)
nin.interval.lo(NA, 1, 3)
```

---

nin.interval.ro	<i>Checks if values are outside of an interval (open on the right)</i>
-----------------	--

---

**Description**

This function checks if the values in the `x` parameter are contained in the interval `[lo, hi)`. NA values are treated as "not in the interval".

**Usage**

```
nin.interval.ro(x, lo, hi)
```

**Arguments**

<code>x</code>	A vector of values. (Lists will be coerced to a numeric vector.)
<code>lo</code>	Left end of the interval.
<code>hi</code>	Right end of the interval.

**Value**

A boolean vector of the same length as `x`.

**See Also**

[in.interval.lo](#), [in.interval.ro](#), [nin.interval.lo](#)

**Examples**

```
nin.interval.ro(c(-1, 0, 1, 2), 0, 1)
nin.interval.ro(NA, 1, 3)
```

---

nlist-deprecated	<i>Smart named list</i>
------------------	-------------------------

---

**Description**

This function is a wrapper around `list()` that assigns names to unnamed arguments based on the unevaluated expression used in the call.

**Usage**

```
nlist(...)
```

**Arguments**

<code>...</code>	List items, possibly named
------------------	----------------------------

**Value**

A named list.

**Author(s)**

Hadley Wickham

**References**

<http://stackoverflow.com/a/5043280/946850>, <http://tolstoy.newcastle.edu.au/R/e9/help/10/03/8392.html>

**See Also**

[list](#)

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

**Examples**

```
a <- 1; b <- 2; c <- 3
nlist(a, b, d=c)
nlist(mean(c(a, b, c)))
```

---

ofactor-deprecated      *Order-preserving factors*

---

**Description**

The function `ofactor()` is a convenience wrapper for `factor()` that orders the levels as they appear in the data if the `levels` argument is not specified.

**Usage**

```
ofactor(x = character(), ...)
```

**Arguments**

`x`                    A vector of data, usually taking a small number of distinct values.  
`...`                Other arguments passed on to `base::factor()`.

**Value**

A factor. See `base::factor()` for details.

**See Also**

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

**Examples**

```
ofactor(3:1)
ofactor(9:12, exclude=11)
identical(ofactor(3:1, levels=1:3), factor(3:1))
```

---

sample.rows-deprecated

*Random Samples and Permutations for Data Frames*

---

**Description**

This function takes a sample of the specified size from the rows of `x` using either with or without replacement.

**Usage**

```
sample.rows(x, size, replace = FALSE, prob = NULL)
```

**Arguments**

<code>x</code>	A data frame.
<code>size</code>	A non-negative integer giving the number of items to choose.
<code>replace</code>	Should sampling be with replacement?
<code>prob</code>	A vector of probability weights for obtaining the rows of the data frame being sampled.

**Details**

This function internally calls [sample.int\(\)](#).

**Value**

A data frame of the same shape as `x`.

**See Also**

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

## Examples

```
set.seed(42)
sample.rows(data.frame(a=c(1,2,3), b=c(4,5,6), row.names=c('a', 'b', 'c')), 10, replace=TRUE)
```

---

seconds.to.hms-deprecated

*Converts a time value given as number of seconds since midnight to the H:M:S format*

---

## Description

This function is very similar to `strftime` with the `%X` conversion specification. Hour values larger than 24 are permitted. Fractions will be rounded down to the next integer. Non-numeric values are coerced to NA with a warning.

## Usage

```
seconds.to.hms(x)
```

## Arguments

x                    A (vector of) numbers.

## Value

A (vector of) character values of the same length as x.

## See Also

[base::strftime\(\)](#)

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [thisfile-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

## Examples

```
seconds.to.hms(c(1, 60, 3600.5))
seconds.to.hms(c(100000, -4000.5))
seconds.to.hms("invalid")
```



---

thisfile-deprecated *Determines the path of the currently running script*

---

### Description

R does not store nor export the path of the currently running script. This is an attempt to circumvent this limitation by applying heuristics (such as call stack and argument inspection) that work in many cases.

### Usage

```
thisfile()  
  
thisfile_source()  
  
thisfile_r()  
  
thisfile_rscript()  
  
thisfile_knit()
```

### Details

This functions currently work only if the script was sourced, processed with knitr, or run with Rscript or using the `--file` parameter to the R executable. For code run with Rscript, the exact value of the parameter passed to Rscript is returned.

### Value

The path of the currently running script, NULL if it cannot be determined.

### Author(s)

Kirill Müller, Hadley Wickham, Michael R. Head

### References

<http://stackoverflow.com/q/1815606/946850>

### See Also

[base::source\(\)](#), [utils::Rscript\(\)](#), [base::getwd\(\)](#)

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [tll-deprecated](#), [vswitch-deprecated](#)

## Examples

```
## Not run: thisfile()
```

---

tll-deprecated	<i>Transposes a list of lists</i>
----------------	-----------------------------------

---

## Description

The argument is assumed to be a list of  $n$  (named) lists with length  $m$  each. It is converted to a (named) list of  $m$  elements with length  $n$  each.

## Usage

```
tll(1)
```

## Arguments

1 List of lists, possibly named.

## Value

A list of lists corresponding to a transposition of the argument.

## See Also

[base::t\(\)](#)

Other deprecated functions: [coalesce.na-deprecated](#), [df\\_to\\_list-deprecated](#), [hms.to.seconds-deprecated](#), [kimisc-deprecated](#), [list\\_to\\_df-deprecated](#), [nc-deprecated](#), [nlist-deprecated](#), [ofactor-deprecated](#), [sample.rows-deprecated](#), [seconds.to.hms-deprecated](#), [thisfile-deprecated](#), [vswitch-deprecated](#)

## Examples

```
tll(list(list(1, 2), list(3, 4)))  
tll(list(list(a=1, b=2), list(a=3, b=4)))  
tll(list(x=list(a=1, b=2), y=list(a=3, b=4)))
```

---

vswitch-deprecated	<i>Vectorized switch</i>
--------------------	--------------------------

---

**Description**

The function `vswitch` is a vectorized version of `base::switch()` optimized for performance.

**Usage**

```
vswitch(EXPR, ...)
```

**Arguments**

EXPR	an expression evaluating to a number or a character string.
...	the list of alternatives. If it is intended that EXPR has a character-string value these will be named, perhaps except for one alternative to be used as a 'default' value.

**Details**

Only the EXPR argument is treated as a vector. In particular, if any of the alternatives (or the default alternative) is a vector, the result will be a list of vectors.

**Value**

The value of one of the elements of `...`, or NA whenever no element is selected. Contrary to `base::switch()` the result is always visible.

**See Also**

Other deprecated functions: `coalesce.na-deprecated`, `df_to_list-deprecated`, `hms.to.seconds-deprecated`, `kimisc-deprecated`, `list_to_df-deprecated`, `nc-deprecated`, `nlist-deprecated`, `ofactor-deprecated`, `sample.rows-deprecated`, `seconds.to.hms-deprecated`, `thisfile-deprecated`, `tll-deprecated`

# Index

assign, [5](#), [6](#)  
assign(), [6](#)

base::cut(), [3](#)  
base::diff(), [7](#)  
base::factor(), [14](#)  
base::format(), [4](#)  
base::getwd(), [17](#)  
base::source(), [17](#)  
base::strftime(), [16](#)  
base::strptime(), [8](#)  
base::switch(), [19](#)  
base::t(), [18](#)

c, [11](#)  
c(), [11](#)  
coalesce.na (coalesce.na-deprecated), [3](#)  
coalesce.na-deprecated, [3](#)  
cut\_format, [3](#)

df\_to\_list (df\_to\_list-deprecated), [4](#)  
df\_to\_list-deprecated, [4](#)

export, [5](#), [6](#)  
export.list, [5](#), [6](#)  
export.list(), [5](#)

factor(), [14](#)

gdiff, [7](#)

hms.to.seconds  
    (hms.to.seconds-deprecated), [8](#)  
hms.to.seconds-deprecated, [8](#)

in.interval.lo, [9](#), [10](#), [12](#), [13](#)  
in.interval.ro, [9](#), [9](#), [12](#), [13](#)

kimisc (kimisc-package), [2](#)  
kimisc-deprecated, [10](#)  
kimisc-package, [2](#)

list, [14](#)  
list(), [13](#)  
list\_to\_df, [4](#)  
list\_to\_df (list\_to\_df-deprecated), [10](#)  
list\_to\_df-deprecated, [10](#)

nc (nc-deprecated), [11](#)  
nc-deprecated, [11](#)  
nin.interval.lo, [9](#), [10](#), [12](#), [13](#)  
nin.interval.ro, [9](#), [10](#), [12](#), [13](#)  
nlist, [11](#)  
nlist (nlist-deprecated), [13](#)  
nlist-deprecated, [13](#)

ofactor (ofactor-deprecated), [14](#)  
ofactor-deprecated, [14](#)

sample.int(), [15](#)  
sample.rows (sample.rows-deprecated), [15](#)  
sample.rows-deprecated, [15](#)  
seconds.to.hms  
    (seconds.to.hms-deprecated), [16](#)  
seconds.to.hms-deprecated, [16](#)

thisfile (thisfile-deprecated), [17](#)  
thisfile-deprecated, [17](#)  
thisfile\_knit (thisfile-deprecated), [17](#)  
thisfile\_r (thisfile-deprecated), [17](#)  
thisfile\_rscript (thisfile-deprecated),  
    [17](#)  
thisfile\_source (thisfile-deprecated),  
    [17](#)  
tll (tll-deprecated), [18](#)  
tll-deprecated, [18](#)

utils::Rscript(), [17](#)

vswitch (vswitch-deprecated), [19](#)  
vswitch-deprecated, [19](#)