

# Package ‘cmrutils’

January 2, 2012

**Type** Package

**Title** Misc Functions of the Center for the Mathematical Research

**Version** 1.2-2

**Date** 2011-11-04

**Author** Andrey Paramonov

**Maintainer** Irakliy Sariev <gilby@bk.ru>

**Depends** R (>= 2.8.1), grDevices, chron

**Imports** grDevices

**Description** A collection of useful helper routines developed by students of the Center for the Mathematical Research, Stankin, Moscow.

**License** GPL (>= 3)

**URL** <http://aparamon.msk.ru/svn/study/R-packages/cmrutils/>

**Repository** CRAN

**Date/Publication** 2011-11-04 11:04:02

## R topics documented:

cmrutils-package . . . . .	2
empty . . . . .	2
makefig . . . . .	3
nan2na . . . . .	4
slice . . . . .	5
time365 . . . . .	6

<b>Index</b>	<b>8</b>
--------------	----------

---

cmrutils-package

*Misc Functions of the Center for the Mathematical Research*

---

### Description

A collection of useful helper routines developed by students of the Center for the Mathematical Research, Stankin, Moscow.

### Details

Package: cmrutils  
Type: Package  
Version: 1.2-2  
Date: 2011-11-04  
License: GPL (>= 3)

### Author(s)

Andrey Paramonov <cmr.Pent@gmail.com>

---

empty

*Helper Routine*

---

### Description

Creates empty copy of object.

### Usage

```
empty(x)
```

### Arguments

x                    An object to create copy from.

### Value

Returns empty (filled with NAs) object having the same structure and attributes as x.

**Examples**

```
empty(0)
empty(1:10)
empty(matrix(1:4, 2, 2))
empty(array(1:24, dim = c(2, 3, 4)))

empty(list(a = 1, b = 2))
empty(data.frame(a = 1:2, b = 3:4))

empty(ts(1:10, frequency = 5))
```

---

makefig

*Helper Plotting Routine*


---

**Description**

Sets up graphical device and plots figure. For PDF device, also takes care of embedding fonts (see Details).

**Usage**

```
makefig(plotfun, ..., device = c('window', 'pdf', 'svg'),
        width = 8, height = 6, scale = pointsize/12, pointsize = 12,
        filename = 'Rplot', family = 'Helvetica')
```

**Arguments**

plotfun	A function which does the actual plotting.
...	Additional arguments passed to plotfun.
device	The type of graphical device.
width, height	The <i>unscaled</i> width and height of the graphics region in inches. Actual size is <code>c(width, height)*scale</code> .
scale	The scaling factor. Is 1 for the default pointsize.
pointsize	The base font size in points.
filename	The filename for PDF or SVG device. If "standard" extension is missing, it is added automagically.
family	The font family for PDF device (see Details).

**Details**

For PDF device, function does font embedding via call to [embedFonts](#). The package provides additional font families "CMRoman", "CMSans" which work nicely with cyrillics. These families are based on CM-Super font package by Vladimir Volovich and CMSYASE symbol font by Paul Murrell.

Both metric (\*.afm) and outline (\*.pfb) files for the "CMRoman", "CMSans" fonts are provided with the package. This allows to embed these fonts even if they are not installed in the system.

**Value**

Returns the result value of call to plotfun *invisibly*.

**See Also**

[x11](#), [pdf](#), [svg](#),

Paul Murrell (2005?) Using Computer Modern Fonts in R Graphics. <http://www.stat.auckland.ac.nz/~paul/R/CM/CMR.html>

**Examples**

```
x <- rnorm(10)

myplot <- function(x, legend, ...)
{
  plot(x, ..., pch = 1)
  legend('bottomright', bg = 'white',
        legend = legend, pch = 1)
}

makefig(myplot, x = x, legend = 'Simulation')
makefig(myplot, x = x, legend = 'Simulation', pointsize = 8)

makefig(myplot, x = x, legend = 'Simulation',
        device = 'pdf', filename = 'embedding.pdf', family = 'CMSans')

mapply(makefig, device = c('window', 'pdf', 'svg'),
       MoreArgs = list(plotfun = myplot, x = x, legend = 'Simulation',
                       filename = 'test', family = 'CMSans'))
```

---

nan2na

*Helper Routine*

---

**Description**

Replaces non-finite values with NAs in array-like objects.

**Usage**

```
nan2na(x)
```

**Arguments**

x                    An array-like object.

**Value**

Returns `x` with non-finite values replaced with NAs.

**See Also**

[is.finite](#), [NA](#).

**Examples**

```
nan2na(Inf)
nan2na(rep(c(0, -Inf, Inf), 3))
nan2na(matrix(c(0, Inf, -Inf, 0), 2, 2))
nan2na(array(c(0, -Inf, Inf, 1, NaN), dim = c(2, 3, 4)))
nan2na(ts(rep(c(0, -Inf, Inf), 2), frequency = 5))
```

---

slice

*Helper Routine*

---

**Description**

Slices array by specified dimension.

**Usage**

```
slice(x, MARGIN, n)
```

**Arguments**

<code>x</code>	An array-like object.
<code>MARGIN</code>	An integer giving the dimension to slice by.
<code>n</code>	Number of slice.

**Value**

Returns array of dimensions `dim(x)[-MARGIN]`. The corresponding `dimnames` are preserved.

**See Also**

[slice.index](#).

**Examples**

```

x <- 1:10
slice(x, 1, 1)

x <- matrix(1:4, 2, 2)
rownames(x) <- c("Row 1", "Row 2")
colnames(x) <- c("Col 1", "Col 2")
x
slice(x, 1, 1)
slice(x, 1, 2)
slice(x, 2, 1)
slice(x, 2, 2)

x <- array(1:24, dim = c(2, 3, 4))
dimnames(x) <- list(letters[1:2], letters[3:5], letters[6:9])
x
slice(x, 1, 1)
slice(x, 2, 1)
slice(x, 3, 1)

```

time365

*Helper Time-Series Routines***Description**

Converts between [Date](#) objects and sampling times of daily time series with the natural time period of 1 year.

**Usage**

```

time365toDate(x)
time365fromDate(x)

```

**Arguments**

`x` A vector of sampling times (`time365toDate`) or dates (`time365fromDate`).

**Details**

A simple approach to modelling daily time series with the natural time period of 1 year is to use [ts](#) object with `frequency = 365`, that is to assume no leap days.

`time365fromDate` returns NA for the leap day February 29.

**Value**

A vector of dates (`time365fromDate`) or sampling times (`time365toDate`).

**See Also**

[time.](#)

**Examples**

```
## non-leap year
x <- ts(1:4, start = c(1991, 58), frequency = 365)
time365toDate(time(x))
d <- seq.Date(as.Date('1991-02-27'), as.Date('1991-03-02'), 1)
time365fromDate(d)

## leap year
x <- ts(1:4, start = c(1992, 58), frequency = 365)
time365toDate(time(x))
d <- seq.Date(as.Date('1992-02-27'), as.Date('1992-03-02'), 1)
time365fromDate(d)

x <- seq(1900, 2100, by = 1/365)
stopifnot(all(abs(x - time365fromDate(time365toDate(x))) <
  getOption("ts.eps"))))
```

# Index

- \*Topic **hplot**
  - makefig, [3](#)
- \*Topic **package**
  - cmrutils-package, [2](#)
- \*Topic **ts**
  - time365, [6](#)
- \*Topic **utilities**
  - cmrutils-package, [2](#)
  - empty, [2](#)
  - makefig, [3](#)
  - nan2na, [4](#)
  - slice, [5](#)
  - time365, [6](#)

cmrutils (cmrutils-package), [2](#)  
cmrutils-package, [2](#)

Date, [6](#)

embedFonts, [3](#)  
empty, [2](#)

is.finite, [5](#)

makefig, [3](#)

NA, [5](#)  
nan2na, [4](#)

pdf, [4](#)

slice, [5](#)  
slice.index, [5](#)  
svg, [4](#)

time, [7](#)  
time365, [6](#)  
time365fromDate (time365), [6](#)  
time365toDate (time365), [6](#)  
ts, [6](#)

x11, [4](#)