Package 'OuhscMunge'

January 10, 2016

Title Data Manipulation Operations

Description Data manipulation operations frequently used in OUHSC BBMC projects.

Version 0.1.6.9000

Date 2016-01-08

Author Will Beasley [aut, cre]

Maintainer Will Beasley <wibeasley@hotmail.com>

URL https://github.com/OuhscBbmc/OuhscMunge, http://ouhsc.edu/bbmc/

BugReports https://github.com/OuhscBbmc/OuhscMunge/issues

License GPL-2

LazyData TRUE

Depends R(>= 3.1.0)

Imports devtools (>= 1.8.0), lubridate

Suggests RODBC, testthat

RoxygenNote 5.0.1

R topics documented:

clump_month_date *Assign date for a given year & month*

Description

This accepts a date, but changes the day. Set/degrade/clump all the days within a month to the same day.

Usage

```
clump_month_date(date_detailed, day_of_month = 15L)
```

Arguments

date_detailed	The Date value containing the desired year and month. The day will be overwritten. Required
day_of_month	The factor label assigned to the missing value. Defaults to 15.

Details

We use this frequently to set/degrade/clump all the days to the middle of their respective month (ie, the 15th day). The midpoint of a month is usually the most appropriate summary location. It makes graphs more intuitive. Using the midpoint of month can also avoid problems with timezones. A date won't get nudged to a neighboring month accidentally.

Value

An array of Date values.

Note

A stop error will be thrown if date_detailed is not a Date, or if day_of_month is not bounded by [1, 31]. Be careful that if you set a November date the 31st day, the result will be December 1st. Consequently, we recommend not setting the day to a value after the 28.

Author(s)

Will Beasley

Examples

```
library(OuhscMunge)
detailed <- seq.Date(from=as.Date("2011-04-21"), to=as.Date("2011-07-14"), by="day")
clumped <- clump_month_date(detailed)
table(clumped)
# 2011-04-15 2011-05-15 2011-06-15 2011-07-15
#         10         31         30         14
```

`headstart_utilities` *Utilities for outputting characteristics of a datatset used it code.*

Description

These functions are used during the execution of a program. Rather they produce snippets that can be pasted into code, and help the developer avoid some typing.

Usage

```
column_rename_headstart( d )
column_class_headstart( d )
column_value_headstart( x )
```

Arguments

`d`	A `data.frame` to describe.
`x`	A vector to describe.

Value

Prints formatted code to the console.

Author(s)

Will Beasley

Examples

```
column_rename_headstart(datasets::OrchardSprays)
column_class_headstart(datasets::OrchardSprays)
column_value_headstart(datasets::OrchardSprays$treatment)
```

`OuhscMunge` *Data manipulation operations frequently used in OUHSC BBMC projects. <http://www.ouhsc.edu/bbmc/>*

Description

Thanks to Funders, including HRSA/ACF D89MC23154

OUHSC CCAN Independent Evaluation of the State of Oklahoma Competitive Maternal, Infant, and Early Childhood Home Visiting (MIECHV) Project., which evaluates MIECHV expansion and enhancement of Evidence-based Home Visitation programs in four Oklahoma counties.

Details

OuhscMunge.

Note

The release version will eventually be available through CRAN by running `install.packages('OuhscMunge')`. The most recent development version is available through GitHub by running `devtools::install_github (repo = 'OuhscBbmc/OuhscMunge')` (make sure devtools is already installed). If you're having trouble with the package, please install the development version. If this doesn't solve your problem, please create a new issue, or email Will.

Author(s)

William Howard Beasley, University of Oklahoma Health Sciences Center, College of Medicine, Dept of Pediatrics, BBMC.

Maintainer: Will Beasley <wibeasley@hotmail.com>

Examples

```
## Not run:
# Install/update REDCapR with the release version from CRAN.
install.packages('OuhscMunge') #But it's not on CRAN yet.

# Install/update REDCapR with the development version from GitHub
#install.packages('devtools') #Uncomment if `devtools` isn't installed already.
devtools::install_github('OuhscBbmc/OuhscMunge')

## End(Not run)
```

`replace_nas_with_explicit`

Create explicit factor level for missing values.

Description

Missing values are converted to a factor level. This explicit assignment can reduce the chances that missing values are inadvertantly ignored. It also allows the presence of a missing to become a predictor in models.

Usage

```
replace_nas_with_explicit(scores, new_na_label = "Unknown",
  create_factor = FALSE, add_unknown_level = FALSE)
```

Arguments

scores	An array of values, ideally either factor or character. Required
new_na_label	The factor label assigned to the missing value. Defaults to Unknown.
create_factor	Converts scores into a factor, if it isn't one already. Defaults to FALSE.
add_unknown_level	Should a new factor level be created? (Specify TRUE if it already exists.) Defaults to FALSE.

Value

An array of values, where the NA values are now a factor level, with the label specified by the new_na_label value.

Note

The create_factor parameter is respected only if scores isn't already a factor. Otherwise, levels without any values would be lost.

A stop error will be thrown if the operation fails to convert all the NA values.

Author(s)

Will Beasley

Examples

```
library(OuhscMunge) #Load the package into the current R session.
missing_indices <- c(3, 6, 8, 25)
# With a character variable:
a <- letters
a[missing_indices] <- NA_character_
a <- OuhscMunge::replace_nas_with_explicit(a)

# With a factor variable:
b <- factor(letters, levels=letters)
b[missing_indices] <- NA_character_
b <- OuhscMunge::replace_nas_with_explicit(b, add_unknown_level=TRUE)
```

Index