Package 'abjutils'

January 19, 2018

Type Package

Date 2018-01-18

Title Useful Tools for Jurimetrical Analysis Used by the Brazilian Jurimetrics Association

Description The Brazilian Jurimetrics Association (ABJ in Portuguese, see <http://www.abjur.org.br/en/> for more information) is a non-profit organization which aims to investigate and promote the use of statistics and probability in the study of Law and its institutions. This package implements general purpose tools used by ABJ, such as functions for sampling and basic manipulation of Brazilian lawsuits identification number. It also implements functions for text cleaning, such as accentuation removal.

Version 0.2.1

Maintainer Caio Lente <ctlente@gmail.com>

URL https://github.com/abjur/abjutils

LazyData TRUE

Depends R (>= 3.1)

License MIT + file LICENSE

Imports stringr, dplyr, httr, tibble, devtools, magrittr, plyr, purrr, rstudioapi, scales, stringi, glue, tidyr, progress

Suggests testthat

RoxygenNote 6.0.1

NeedsCompilation no

Author Julio Trecenti [aut], Athos Damiani [ctb], Fernando Correa [aut], Caio Lente [aut, cre], Brazilian Jurimetrics Association [cph]

Repository CRAN

Date/Publication 2018-01-19 18:26:56 UTC

R topics documented:

`build_id` *Add separators to lawsuit IDs*

Description

Add separators to lawsuit IDs

Usage

```
build_id(id)
```

Arguments

`id`	One or more lawsuit IDs

`calc_dig` *Calculate digits for Brazilian lawsuit identification numbers*

Description

Returns the check digit of a lawsuit numbers in the format unified by the brazillian National Council of Justice.

Usage

```
calc_dig(num, build = FALSE)
```

Arguments

num	Ordered digits of the lawsuit number (including 0's) excluding the check digit
build	Whether or not the function return the complete lawsuit number (or only the check digits)?

Value

The check digits or the complete identification number

Examples

```
{
calc_dig("00104062001826004", build = TRUE)
calc_dig("00104062001826004", build = FALSE)

#will fail
## Not run:
calc_dig("00104062001826000", build = TRUE)

## End(Not run)
}
```

---

carefully *Vectorized, parallel, safe and verbose function factory*

Description

Wraps a function so that iterating over a set of inputs is easily parallelizable, and interruption-free.

Usage

```
carefully(.f, p = 0.05, cores = 1)
```

Arguments

.f	Function to be wrapped
p	Probability of function printing the index of the input it's currently processing
cores	Number of cores to use when iterating over vectorized inputs

Examples

```
## Not run:
# Function that takes a string and pastes two other strings
# a its beginning and end respectivelly
pad <- function(str, b = "", a = "") { paste0(b, str, a) }

# Create wrapped version of pad() that executes over 4 cores,
```

```
# captures errors, and prints its current iteration with a
# probability of 50%
new_pad <- carefully(pad, p = 0.5, cores = 4)

# Execute new_pad() with some sample data
new_pad(c("asdf", "poiu", "qwer"), b = "0", a = "1")

## End(Not run)
```

`check_dig` *Validate check digits for Brazilian lawsuits identification number*

Description

Verifies if a check digit is correct

Usage

```
check_dig(num)
```

Arguments

`num`	String containing the complete lawsuit number

Value

Whether or not the check digit is well calculated

Examples

```
{
check_dig("0005268-75.2013.8.26.0100")

## Not run:
check_dig("0005268-75.2013.8.26.100", build = TRUE)

## End(Not run)
}
```

clean_id *Remove separators from lawsuit IDs*

Description

Remove separators from lawsuit IDs

Usage

```
clean_id(id)
```

Arguments

id	One or more lawsuit IDs

dvec *Vectorize functions (DEPRECATED)*

Description

Iterate a function and wrap a `dplyr::failwith()` around it.

Usage

```
dvec(fun, itens, ..., verbose = TRUE, p = 0.05)
```

Arguments

fun	Function to be iterated
itens	Character vector of inputs
...	Other parameters for `fun`
verbose	Should dvec print the current `item` (if `TRUE`, shows a message with probability `p`)?
p	Probability of printing a message

escape_unicode *Escape accented characters in a document*

Description

This function is used by the "Escape Unicode" add-in and removes all accented characters from the current file, replacing them by their equivalent Unicode-escaped values.

Usage

```
escape_unicode()
```

extract_parts *Extract different parts from lawsuit ID*

Description

Given one or more lawsuit IDs, this function extracts one or more parts of the IDs given the following correspondence:

- "N": number
- "D": verification digits
- "A": year
- "J": segment
- "T": court
- "O": origin
- "": all of the above

Usage

```
extract_parts(id, parts = "")
```

Arguments

id	One or more lawsuit IDs
parts	String or string vector with desired parts (see **description**)

Examples

```
{
extract_parts("00104062001826004", "N")
extract_parts("00104062001826004", c("N", "A", "O"))
}
```

lsos *Improved list of objects*

Description

Elegantly list objects in a R session.

Usage

```
lsos(pos = 1, pattern, order.by = "Size", decreasing = TRUE,
  head = TRUE, n = 10)
```

Arguments

pos	Where to look for the object (see "Details" in base::get()'s documentation)
pattern	An optional regular expression to match names (utils::glob2rx() can be used to convert wildcard patterns to regular expressions)
order.by	Sort by "Size" (default), "Type", "Rows" or "Columns"
decreasing	Should the sorting be decreasing?
head	Should utils::head() function be used for printing?
n	How many lines utils::head() function should show?

References

http://stackoverflow.com/questions/1358003/tricks-to-manage-the-available-memory-in-an-r-session

---

precision *Mirror of scales:::precision()*

Description

Mirror of scales:::precision()

Usage

```
precision(x)
```

Arguments

x	See scales:::precision()

---

rm_accent *Remove accentuation*

Description

Remove accented characters from strings converting them to ASCII.

Usage

```
rm_accent(x)
```

Arguments

x	A string vector

Value

A version of x without non-ASCII characters

rm_accent_from_names *Remove accentuation from column names (DEPRECATED)*

Description

Remove accented characters from column names converting them to ASCII.

Usage

```
rm_accent_from_names(dat)
```

Arguments

dat	A dataset

Value

A version of dat without non-ASCII characters.

sample_cnj *Generate sample Brazilian lawsuit identification numbers*

Description

Returns a data frame containing a random sample of lawsuit numbers distributed according to some regional and jurisdictional parameters. The implementation supports both vector and scalar parameters, depending whether or not the function should uniformly sample from a scope of lawsuit numbers or one should define the parameters for each sample unit.

Usage

```
sample_cnj(n, foros, anos, orgao, tr, first_dig = "0", sample_pars = TRUE,
  return_df = TRUE)
```

Arguments

n	A non negative integer giving the number of codes to generate
foros	One or more strings with 4 characters indicating the juridical forum for the sampled codes
anos	One or more strings with 4 characters indicating the distribution years of the generated codes
orgao	One or more strings with 1 character indicating the jurisdiction of the sampled codes.
tr	One or more strings with 1 character indicating the court of the generated codes
first_dig	The first digit of the lawsuit code ("0" by default and sampled if "")
sample_pars	Whether or not the parameters define the characteristics of the codes
return_df	Whether or not the function should return a data frame

Value

A data frame or a vector containing a random sample of lawsuits IDs

Examples

```
{
#sampling the parameters
sample_cnj(3, foros = "0000",
anos = "2015", orgao = 8, tr = 26,
first_dig = "0",sample_pars = TRUE, return_df = FALSE)

sample_cnj(10, foros = c("0000","0001"),
anos = c("2014","2015"), orgao = 8, tr = 26,
first_dig = "0",sample_pars = TRUE, return_df = FALSE)

#not sampling the parameters

sample_cnj(3, foros = c("0000","0001","0002"),
anos = c("2014","2015","2016"), orgao = rep(8,3), tr = rep(26,3),
first_dig = "0",sample_pars = FALSE, return_df = FALSE)
}
```

---

separate_cnj *Separate a lawsuit ID column into its parts*

Description

Wrapper around `tidyr::separate()` that splits a column with lawsuit IDs into 6 columns with its parts (see `extract_parts()`). Note that the IDs must be built (see `build_id()`).

Usage

```
separate_cnj(data, col, ...)
```

Arguments

`data`	A data frame
`col`	Column name or position (see `tidyr::separate()`)
`...`	Other arguments passed on to `tidyr::separate()`

tabela *Produce frequency and relative frequency tables*

Description

Produces a contingency table of the elements of a vector calculating relative frequencies as well.

Usage

```
tabela(x, label = "variavel")
```

Arguments

x	A vector
label	Quoted name of the column to create in output

Value

A data frame containing frequency and relative frequencies for the levels of x

use_pipe *Add pipe template*

Description

Adds pipe template to package documentation.

Usage

```
use_pipe(pkg = ".")
```

Arguments

pkg	Package description (can be path or package name)

Index