Package ‘jmvcore’

May 14, 2020

Type  Package
Title  Dependencies for the 'jamovi' Framework
Version  1.2.19
Date  2020-05-14
Author  Jonathon Love
Maintainer  Jonathon Love <jon@thon.cc>
Description  A framework for creating rich interactive analyses for the jamovi platform (see <https://www.jamovi.org> for more information).
URL  https://www.jamovi.org
BugReports  https://github.com/jamovi/jmvcore/issues
License  GPL (>= 2)
ByteCompile  yes
Depends  R (>= 3.2)
Imports  R6 (>= 1.0.1), rlang (>= 0.3.0.1), jsonlite, base64enc, stringi
Suggests  testthat (>= 1.0.2), RProtoBuf, knitr, ggplot2, RColorBrewer
RoxygenNote  6.1.1
NeedsCompilation  no
Repository  CRAN
Date/Publication  2020-05-14 05:50:02 UTC

R topics documented:

Analysis ......................................................... 2
canBeNumeric ................................................. 3
Cell.BEGIN_GROUP ............................................. 3
colorPalette ................................................... 4
composeFormula ................................................. 5
composeTerm .................................................... 5
### Analysis

**Analysis**

**Description**

the `jmvcore` Object classes

**Usage**

- **Array**
  - **Column**
  - **Group**
  - **Html**
  - **Image**
Preformatted
State
Table

Format
An object of class \texttt{R6ClassGenerator} of length 25.

<table>
<thead>
<tr>
<th>canBeNumeric</th>
<th>Determines whether an object is or can be converted to numeric</th>
</tr>
</thead>
</table>

Description
Determines whether an object is or can be converted to numeric

Usage
canBeNumeric\(\text{object}\)

Arguments
\begin{itemize}
\item \text{object} \quad \text{the object}
\end{itemize}

\begin{itemize}
\item \text{Cell.BEGIN\_GROUP} \quad \text{Constants to specify formatting of Table cells}
\end{itemize}

Description
Cell.BEGIN\_GROUP adds spacing above a cell

Usage
\begin{itemize}
\item \text{Cell.BEGIN\_GROUP}
\item \text{Cell.END\_GROUP}
\item \text{Cell.BEGIN\_END\_GROUP}
\item \text{Cell.NEGATIVE}
\item \text{Cell.INDENTED}
\end{itemize}
Format
An object of class numeric of length 1.

Details
Cell.END_GROUP add spacing below a cell
Cell.BEGIN_END_GROUP add spacing above and below a cell
Cell.NEGATIVE specifies that the cells contents is negative

Examples
## Not run:

table$addFormat(rowNo=1, col=1, Cell.BEGIN_END_GROUP)

## End(Not run)

colorPalette

A function that creates a color palette

Description
A function that creates a color palette

Usage
colorPalette(n = 5, pal = "jmv", type = "fill")

Arguments
n Number of colors needed
pal Color palette name
type 'fill' or 'color'

Value
a vector of hex color codes
composeFormula

Compose a formula string

Description

Compose a formula string

Usage

composeFormula(lht, rht)

Arguments

lht list of character vectors making up the left
rht list of character vectors making up the right

Value

a string representation of the formula

Examples

composeFormula(list('a', 'b', c('a', 'b')))
# ~a+b+a:b

composeFormula('f', list('a', 'b', c('a', 'b')))
# "f~a+b+a:b"

composeFormula('with spaces', list('a', 'b', c('a', 'b')))
'~a+b+a:b'

composeTerm

Compose and decompose interaction terms to and from their components

Description

Compose and decompose interaction terms to and from their components
Usage

composeTerm(components)
composeTerms(listOfComponents)
decomposeTerm(term)
decomposeTerms(terms)

Arguments

components   a character vectors of components
listOfComponents  a list of character vectors of components
term   a string with components separated with colons
terms   a character vector of components separated with colons

Examples

composeTerm(c('a', 'b', 'c'))
# 'a:b:c'
composeTerm(c('a', 'b', 'with space'))
# 'a:b:with space'
decomposeTerm('a:b:c')
# c('a', 'b', 'c')
decomposeTerm('a:b:with space')
# c('a', 'b', 'with space')

constructFormula  Construct a formula string

Description

Construct a formula string

Usage

constructFormula(dep = NULL, terms)

Arguments

dep   the name of the dependent variable
terms   list of character vectors making up the terms
Value

a string representation of the formula

Examples

```r
constructFormula(terms=list('a', 'b', c('a', 'b')))
# a+b+a:b

constructFormula('f', list('a', 'b', c('a', 'b')))
# "f=a+b+a:b"

constructFormula('with spaces', list('a', 'b', c('a', 'b')))
'\textbackslash{}~a+b+a:b'
```

create  

Create an analysis

Description

Used internally by jamovi

Usage

```r
create(ns, name, optionsPB, datasetId, analysisId, revision)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ns</td>
<td>package name</td>
</tr>
<tr>
<td>name</td>
<td>analysis name</td>
</tr>
<tr>
<td>optionsPB</td>
<td>options protobuf object</td>
</tr>
<tr>
<td>datasetId</td>
<td>dataset id</td>
</tr>
<tr>
<td>analysisId</td>
<td>analysis id</td>
</tr>
<tr>
<td>revision</td>
<td>revision</td>
</tr>
</tbody>
</table>
createError  
Create and throw errors

Description

These functions are convenience functions for creating and throwing errors.

Usage

createError(formats, code = NULL, ...)
reject(formats, code = NULL, ...)

Arguments

- **formats**: a format string which is passed to `format`
- **code**: an error code
- **...**: additional arguments passed to `format`

------

decomposeFormula  
Decompose a formula

Description

Decompose a formula

Usage

decomposeFormula(formula)

Arguments

- **formula**: the formula to decompose

Value

a list of lists of the formulas components
enquo

rlang::enquo Simplifies things so packages overriding Analysis don’t need to have rlang in their imports. This is intended for use by classes overriding Analysis

Description

rlang::enquo Simplifies things so packages overriding Analysis don’t need to have rlang in their imports. This is intended for use by classes overriding Analysis

Usage

enquo(arg)

Arguments

arg the argument to enquote

Value

the quosure

extractErrorMessage

Extracts the error message from an error object

Description

Extracts the error message from an error object

Usage

extractErrorMessage(error)

Arguments

error an error object
**format**

*Format a string with arguments*

**Description**

Substitutes the arguments into the argument `str`. See the examples below.

**Usage**

```r
format(str, ..., context = "normal")
```

**Arguments**

- `str`: the format string
- `...`: the arguments to substitute into the string
- `context`: 'normal' or 'R'

**Value**

the resultant string

**Examples**

```r
jmvcore::format("the {} was delish", 'fish')
# 'the fish was delish'

jmvcore::format("the {} was more delish than the {}", 'fish', 'cow')
# 'the fish was more delish than the cow'

jmvcore::format("the {} was more delish than the {}", 'fish', 'cow')
# 'the cow was more delish than the fish'

jmvcore::format("the {0} and the {1}" , which='fish', what='cow')
# 'the cow and the fish'

jmvcore::format("that is simply not {}", TRUE)
# 'that is simply not true'

jmvcore::format("that is simply not {}", TRUE, context='R')
# 'that is simply not TRUE'
```
**isError**

Determine if an object is an error

**Description**

Determine if an object is an error

**Usage**

isError(object)

**Arguments**

object: the object to test

**Value**

TRUE if the object is an error

---

**marshalData**

Marshal the data from an environment into a data frame

**Description**

Marshal the data from an environment into a data frame

**Usage**

marshalData(env, ...)

**Arguments**

env: the environment to marshal from

...: the variables to marshal

**Value**

a data frame
marshalFormula  
*Marshal a formula into options*

**Description**

Marshal a formula into options

**Usage**

```r
marshalFormula(formula, data, from = "rhs", type = "vars",
               permitted = c("numeric", "factor"), subset = ":", required = FALSE)
```

**Arguments**

- **formula**: the formula
- **data**: a data frame to marshal the data from
- **from**: 'rhs' or 'lhs', which side of the formula should be marshalled
- **type**: 'vars' or 'terms', the type of the option be marshalled to
- **permitted**: the types of data the option permits
- **subset**: a subset of the formula to marshal
- **required**: whether this marshall is required or not

---

matchSet  
*Determines the index where an item appears*

**Description**

Determines the index where an item appears

**Usage**

```r
matchSet(x, table)
```

**Arguments**

- **x**: the item to find
- **table**: the object to search

**Value**

the index of where the item appears, or -1 if it isn’t present
Description

removes all rows from the data frame which contain missing values (NA)

Usage

naOmit(object)

Arguments

object the object to remove missing values from

Details

this function is equivalent to na.omit from the stats package, however it preserves attributes on columns in data frames

Options

The jmv Options classes

Description

The jmv Options classes

Usage

Options
  OptionBool
  OptionList
  OptionNMXList
  OptionVariables
  OptionTerm
  OptionVariable
  OptionTerms
  OptionInteger
OptionNumber
OptionString
OptionLevel
OptionGroup
OptionSort
OptionArray
OptionPairs

Format

An object of class R6ClassGenerator of length 25.

| resolveQuo | Evaluates a quosure This is intended for use by classes overriding Analysis |

Description

Evaluates a quosure This is intended for use by classes overriding Analysis

Usage

resolveQuo(quo)

Arguments

quo the quosure to evaluate

Value

the value of the quosure
select

Create a new data frame with only the selected columns

Description

Shorthand equivalent to `subset(df, select=columnNames)`, however it additionally preserves attributes on the columns

Usage

`select(df, columnNames)`

Arguments

- `df`: the data frame
- `columnNames`: the names of the columns to make up the new data frame

Value

the new data frame

sourcify

Converts basic R object into their source representation

Description

Converts basic R object into their source representation

Usage

`sourcify(object, indent = "")`

Arguments

- `object`: the object to convert to source
- `indent`: the level of indentation to use

Value

a string of the equivalent source code
startsWith

Test whether strings start or end with a particular string

Description

Same as base::startsWith() and base::endsWith() except available for R < 3.3

Usage

startsWith(x, prefix)
endsWith(x, suffix)

Arguments

x a string to test
prefix a string to test the presence of
suffix a string to test the presence of
stringifyTerm

Converts a term into a string

Description

Converts a term (a vector of components) into a string for display purposes

Usage

stringifyTerm(components, sep =getOption("jmvTermSep", ":"), raise = FALSE)

Arguments

components a character vector of components
sep a separator to go between the components
raise whether duplicates should be raised to powers

Value

the components joined together into a string for display

Examples

stringifyTerm(c("a", "b", "c"))
# "a:b:c"

stringifyTerm(c("a", "b", "c"), sep=" * ")
# "a * b * c"

options("jmvTermSep", " * ")
stringifyTerm(c("a", "b", "c"))
# "a * b * c"

#" stringifyTerm(c("\"quoted\"", "b", "c"))
# "quoted * b * c"
theme_default  

Description

Creates the default jmv ggplot2 theme

Usage

theme_default(base_size = 16, scale = "none", palette = "jmv")

Arguments

base_size  
Font size
scale  
'none' or 'discrete'
palette  
Color palette name

Value

the default jmv ggplot2 theme

theme_hadley  

Description

Creates the hadley jmv ggplot2 theme

Usage

theme_hadley(base_size = 16, scale = "none", palette = "jmv")

Arguments

base_size  
Font size
scale  
'none' or 'discrete'
palette  
Color palette name

Value

the hadley jmv ggplot2 theme
### theme_min

*Creates the minimal jmv ggplot2 theme*

**Description**

Creates the minimal jmv ggplot2 theme

**Usage**

```r
tHEME_MIN(base_size = 16, scale = "none", palette = "jmv")
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>base_size</td>
<td>Font size</td>
</tr>
<tr>
<td>scale</td>
<td>'none' or 'discrete'</td>
</tr>
<tr>
<td>palette</td>
<td>Color palette name</td>
</tr>
</tbody>
</table>

**Value**

the minimal jmv ggplot2 theme

### theme_spss

*Creates the spss jmv ggplot2 theme*

**Description**

Creates the spss jmv ggplot2 theme

**Usage**

```r
tHEME_SPSS(base_size = 16, scale = "none", palette = "jmv")
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>base_size</td>
<td>Font size</td>
</tr>
<tr>
<td>scale</td>
<td>'none' or 'discrete'</td>
</tr>
<tr>
<td>palette</td>
<td>Color palette name</td>
</tr>
</tbody>
</table>

**Value**

the spss jmv ggplot2 theme
**toB64**

*Convert names to and from Base64 encoding*

**Description**

Note: uses the . and _ characters rather than + and / allowing these to be used as variable names

**Usage**

```r
toB64(names)
fromB64(names)
```

**Arguments**

- `names` the names to be converted base64

---

**toNumeric**

*Converts a vector of values to numeric*

**Description**

Similar to `as.numeric`, however if the object has a values attribute attached, these are used as the numeric values

**Usage**

```r
toNumeric(object)
```

**Arguments**

- `object` the vector to convert
tryNaN

tryNaN (expr)

**Arguments**

expr 

**Value**

the result, or NaN on failure

**Description**

if the expression fails, NaN is returned silently

**Usage**

tryNaN(expr)
Index

*Topic datasets
  Analysis, 2
  Cell.BEGIN_GROUP, 3
  Options, 13
Analysis, 2
Array(Analysis), 2
as.numeric, 20
canBeNumeric, 3
Cell.BEGIN_END_GROUP
  (Cell.BEGIN_GROUP), 3
Cell.BEGIN_GROUP, 3
Cell.END_GROUP (Cell.BEGIN_GROUP), 3
Cell.INDENTED (Cell.BEGIN_GROUP), 3
colorPalette, 4
Column (Analysis), 2
composeFormula, 5
composeTerm, 5
decomposeFormulas (composeTerm), 5
decomposeTerms (composeTerm), 5
endsWith (startsWith), 16
enquo, 9
extractErrorMessage, 9
format, 8, 10
toB64 (fromB64), 20
Group (Analysis), 2
Html (Analysis), 2
Image (Analysis), 2
isError, 11
marshalData, 11
marshalFormula, 12
matchSet, 12
na.omit, 13
naOmit, 13
OptionArray (Options), 13
OptionBool (Options), 13
OptionGroup (Options), 13
OptionInteger (Options), 13
OptionLevel (Options), 13
OptionList (Options), 13
OptionNMXList (Options), 13
OptionNumber (Options), 13
OptionPairs (Options), 13
Options, 13
OptionSort (Options), 13
OptionString (Options), 13
OptionTerm (Options), 13
OptionTerms (Options), 13
OptionVariable (Options), 13
OptionVariables (Options), 13
Preformatted (Analysis), 2
reject (createError), 8
resolveQuo, 14
select, 15
sourcify, 15
startsWith, 16
State (Analysis), 2
stringifyTerm, 17
subset, 15
Table (Analysis), 2
theme_default, 18
theme_hadley, 18
INDEX

theme_min, 19
theme_spss, 19
toB64, 20
toNumeric, 20
tryNaN, 21