Package: PerseusR (via r-universe)

March 24, 2025

Title Perseus R Interop

Version 0.3.4

Author Jan Rudolph <rudolph@biochem.mpg.de>

Maintainer Jan Rudolph <rudolph@biochem.mpg.de>

Description Enables the interoperability between the Perseus platform for omics data analysis (Tyanova et al. 2016) <doi:10.1038/nmeth.3901> and R. It provides the foundation for developing and running Perseus plugins implemented in R by providing all required input and output handling, including data and parameter parsing as described in Rudolph and Cox 2018 <doi:10.1101/447268>.

Depends R (>= 3.3.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

Imports utils, plyr, methods, XML, Biobase, stringr

Suggests testthat, roxygen2, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Date/Publication 2018-11-05 15:25:14 UTC

Additional_repositories https://cranhaven.r-universe.dev

Config/pak/sysreqs libicu-dev libxml2-dev

Repository https://cranhaven.r-universe.dev

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/PerseusR

RemoteSha 1746e1b3018a85b56783f598b6d4dd9db5a31555

RemoteSubdir PerseusR

Contents

annotCols . . . 2
annotCols<- . . . 3
annotRows . . . 4
annotRows<- . . . 4
as.ExpressionSet.matrixData . . . 5
as.matrixData.ExpressionSet . . . 6
boolParamValue . . . 6
create_annotRows . . . 7
description . . . 7
description<- . . . 8
imputeData . . . 9
imputeData<- . . . 9
infer_perseus_annotation_types . . . 10
initialize,matrixData-method . . . 11
intParamValue . . . 11
main . . . 12
main<- . . . 12
matrixData . . . 13
matrixData-class . . . 14
MatrixDataCheck . . . 14
names,matrixData-method . . . 15
names.matrixData . . . 16
parseParameters . . . 16
qualityData . . . 17
qualityData<- . . . 17
read.perseus.default . . . 18
singleChoiceParamInd . . . 19
singleChoiceParamValue . . . 20
write.perseus . . . 21

Index **23**

`annotCols` *Get annotation columns*

Description

Get annotation columns

Usage

```
annotCols(mdata)
```

Arguments

`mdata`	matrixData

See Also

Other matrixData basic functions: annotCols<-, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))
annotCols(mdata)
```

---

annotCols<- *Set annotation columns*

Description

Set annotation columns

Usage

```
annotCols(mdata) <- value
```

Arguments

mdata	matrixData
value	value

See Also

Other matrixData basic functions: annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))
value <- data.frame(d=c('d', 'e', 'f'))
annotCols(mdata) <- value
```

annotRows *Get annotation rows*

Description

Get annotation rows

Usage

```
annotRows(mdata)
```

Arguments

mdata	matrixData

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))
annotRows(mdata)
```

annotRows<- *Set annotation rows*

Description

Set annotation rows

Usage

```
annotRows(mdata) <- value
```

Arguments

mdata	matrixData
value	value

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))
value <- data.frame(y=factor(c('2','2')))
annotRows(mdata) <- value
```

---

as.ExpressionSet.matrixData

Coerces a MatrixData into an ExpressionSet

Description

Coerces a MatrixData object into an ExpressionSet object

Usage

```
as.ExpressionSet.matrixData(mdata)
```

Arguments

mdata	a matrixData object

Details

function to convert a matrixData ExpressionSet

Value

returns an ExpressionSet object

Examples

```
mD <- matrixData(
main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(b=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))

eSet <- as(mD, "ExpressionSet")
print(eSet)
```

as.matrixData.ExpressionSet *Coerces an ExpressionSet into a MatrixData*

Description

function to convert an ExpressionSet object into a matrixData

Usage

```
as.matrixData.ExpressionSet(ExpressionSet)
```

Arguments

ExpressionSet	an ExpressionSet object

Value

returns a matrixData object

Examples

```
eSet <- eSet <- Biobase::ExpressionSet(matrix(1:10, ncol = 2))
mD <- as(eSet, "matrixData")
print(mD)
```

boolParamValue *Bool parameter value*

Description

Extract the value chosen in an BoolParam

Usage

```
boolParamValue(parameters, name)
```

Arguments

parameters	The parameters object (see parseParameters)
name	The name of the parameter

Value

The selected boolean

Examples

```
tmp <- tempfile(fileext = ".xml")
write('<BoolParam Name="test_bool">\n<Value>false</Value>\n</BoolParam>', file=tmp)
parameters <- parseParameters(tmp)
boolParamValue(parameters, "test_bool")
```

---

create_annotRows *Create annotation rows*

Description

Create the annotation rows data.frame from the list of comment rows parsed from the input file and the main columns indicator

Usage

```
create_annotRows(commentRows, isMain)
```

Arguments

commentRows	list of comment rows
isMain	logical array indicating all main columns

See Also

used by read.perseus

---

description *Get column description*

Description

Get column description

Usage

```
description(mdata)
```

Arguments

mdata	matrixData

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))),
description=c('aaa', 'bbb', 'ccc'))
description(mdata)
```

`description<-` *Set column description*

Description

Set column description

Usage

```
description(mdata) <- value
```

Arguments

`mdata`	matrixData
`value`	value

See Also

Other matrixData basic functions: `annotCols<-`, `annotCols`, `annotRows<-`, `annotRows`, `description`, `imputeData<-`, `imputeData`, `main<-`, `main`, `matrixData-class`, `matrixData`, `names,matrixData-method`, `qualityData<-`, `qualityData`

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))
value <- c('aaa', 'bbb', 'ccc')
description(mdata) <- value
```

imputeData *Get imputation of main data frame*

Description

Get imputation of main data frame

Usage

```
imputeData(mdata)
```

Arguments

mdata	matrixData

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))),
imputeData=data.frame(impute=c('False', 'True', 'False')))
imputeData(mdata)
```

imputeData<- *Set imputation of main data frame*

Description

Set imputation of main data frame

Usage

```
imputeData(mdata) <- value
```

Arguments

mdata	matrixData
value	value

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))),
imputeData=data.frame(impute=c('False', 'True', 'False')))
value <- data.frame(impute=c('True', 'True', 'True'))
imputeData(mdata) <- value
```

---

infer_perseus_annotation_types

Infer Perseus type annotation row from DataFrame column classes

---

Description

Infer Perseus type annotation row from DataFrame column classes

Usage

```
infer_perseus_annotation_types(df, typeMap)
```

Arguments

df	The data.frame
typeMap	A list with elements 'Perseus' and 'R'. The ordering determines the mapping

Value

A vector with perseus type annotations

See Also

Based on mapvalues

initialize,matrixData-method *matrixData initializer*

Description

Initializes the annotCols data frame to have the same number of rows as the main data. This might not be the cleanest solution.

Usage

```
## S4 method for signature 'matrixData'
initialize(.Object, ...)
```

Arguments

.Object	Initialized object
...	Additional arguments

intParamValue *Int parameter value*

Description

Extract the value chosen in an IntParam

Usage

```
intParamValue(parameters, name)
```

Arguments

parameters	The parameters object (see parseParameters)
name	The name of the parameter

Value

The selected number

Examples

```
tmp <- tempfile(fileext = ".xml")
write('<IntParam Name="test_int">\n<Value>2</Value>\n</IntParam>', file=tmp)
parameters <- parseParameters(tmp)
intParamValue(parameters, "test_int")
```

main *Get main columns*

Description

Gets the main columns (main matrix) of a matrixData object as a data.frame object

Usage

```
main(mdata)
```

Arguments

mdata	matrixData

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main<-, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))
main(mdata)
```

main<- *Set main columns*

Description

Set main columns

Usage

```
main(mdata) <- value
```

Arguments

mdata	matrixData
value	value

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))))
value<-data.frame(c=c(0,0,0), d=c(1,1,1))
main(mdata) <- value
```

---

matrixData *matrixData constructor*

Description

matrixData constructor

Usage

```
matrixData(...)
```

Arguments

...	main, annotCols, annotRows, description, imputeData, qualityData

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, names,matrixData-method, qualityData<-, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))),
description=c('aaa', 'bbb', 'ccc'),
imputeData=data.frame(impute=c('False', 'True', 'False')),
qualityData=data.frame(quality=c('0', '1', '0')))
```

`matrixData-class` *MatrixData*

Description

MatrixData

Slots

`main` Main expression `data.frame`.

`annotCols` Annotation Columns `data.frame`.

`annotRows` Annotation Rows `data.frame`.

`description` Column descriptions.

`imputeData` Imputation `data.frame`.

`qualityData` Quality values `data.frame`.

See Also

Other matrixData basic functions: `annotCols<-`, `annotCols`, `annotRows<-`, `annotRows`, `description<-`, `description`, `imputeData<-`, `imputeData`, `main<-`, `main`, `matrixData`, `names,matrixData-method`, `qualityData<-`, `qualityData`

`MatrixDataCheck` *MatrixDataCheck: a function to check the validity of an object as a perseus data frame*

Description

Check perseus compatibility of an object

Usage

```
MatrixDataCheck(object, ...)

## Default S3 method:
MatrixDataCheck(object = NULL, main, annotationRows,
  annotationCols, descriptions, imputeData, qualityData, all_colnames, ...)

## S3 method for class 'matrixData'
MatrixDataCheck(object, ...)

## S3 method for class 'list'
MatrixDataCheck(object, ...)

## S3 method for class 'ExpressionSet'
MatrixDataCheck(object, ...)
```

Arguments

object	object to check consistency with perseus data frames
...	additional arguments passed to the respective method
main	Main Data frame
annotationRows	Rows containing annotation information
annotationCols	Columns containing annotation information
descriptions	Descriptions of all the columns
imputeData	Is imputed or not
qualityData	quality number
all_colnames	The colnames to be used

Value

a logical indicating the validity of the object (or series of objects) as a perseus DF or the string of errors

NULL

NULL

NULL

Examples

```
require(PerseusR)

mat <- matrixData(
    main=data.frame(a=1:3, b=6:8),
    annotCols=data.frame(c=c('a','b','c')),
    annotRows=data.frame(x=factor(c('1','1'))))

MatrixDataCheck(mat)
```

---

names,matrixData-method

Get names

Description

Get the column names of main and annotation columns.

Usage

```
## S4 method for signature 'matrixData'
names(x)
```

Arguments

x	matrixData

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, qualityData<-, qualityData

---

names.matrixData *Column names of main and annotation columns*

Description

Column names of main and annotation columns

Usage

```
## S3 method for class 'matrixData'
names(x)
```

Arguments

x	matrixData

---

parseParameters *Parse parameters*

Description

Parse parameters from the parameters xml file.

Usage

```
parseParameters(paramFile)
```

Arguments

paramFile	Parameters xml file

Examples

```
tmp <- tempfile(fileext = ".xml")
write('<IntParam Name="test_int">\n<Value>2</Value>\n</IntParam>', file=tmp)
parameters <- parseParameters(tmp)
```

qualityData *Get quality values of main data frame*

Description

Get quality values of main data frame

Usage

```
qualityData(mdata)
```

Arguments

mdata	matrixData

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData<-

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))),
qualityData=data.frame(quality=c('1', '1', '1')))
qualityData(mdata)
```

qualityData<- *Set quality values of main data frame*

Description

Set quality values of main data frame

Usage

```
qualityData(mdata) <- value
```

Arguments

mdata	matrixData
value	value

See Also

Other matrixData basic functions: annotCols<-, annotCols, annotRows<-, annotRows, description<-, description, imputeData<-, imputeData, main<-, main, matrixData-class, matrixData, names,matrixData-method, qualityData

Examples

```
mdata <- matrixData(main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(c=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))),
qualityData=data.frame(quality=c('1', '1', '1')))
value <- data.frame(quality=c('0', '0', '0'))
qualityData(mdata) <- value
```

---

read.perseus.default *Read Perseus matrix files*

Description

Read the custom Perseus matrix file format *.txt into R.

Usage

```
read.perseus.default(con, check = TRUE, additionalMatrices = FALSE)

read.perseus.as.list(con, check = TRUE)

read.perseus.as.matrixData(con, check = TRUE,
  additionalMatrices = FALSE)

read.perseus.as.ExpressionSet(con, check = TRUE)

read.perseus(con, check = TRUE, additionalMatrices = FALSE)
```

Arguments

con	A connection object or the path to input file
check	Logical indicating whether to check for the validity of the exported object (slightly slower)
additionalMatrices	Logical indication whether to write out quality and imputation matrices in perseus format

Value

Defaults to a matrixData object.

Functions

- `read.perseus.default`: Returns a list used internally to generate all other outputs
- `read.perseus.as.list`: Returns explicitly as a list
- `read.perseus.as.matrixData`: Returns explicitly as a specialized matrix data object
- `read.perseus.as.ExpressionSet`: Returns a bioconductor expression set object

Note

Limitations to column names in R still apply. Column names valid in Perseus, such as 'Column 1' will be changed to 'Column.1'

If the provided connection `con` is a character string, it will assumed to be a file path. A `connection` which is not seekable (see `isSeekable`) will be written to a temporary file. Any connection will be closed when `read.perseus` exits. `read.perseus.as.list`, `read.perseus.as.matrixData` and `read.perseus.as.ExpressionSet` are also available depending on the class desired as an output

See Also

`write.perseus`

`matrixData`

Examples

```
tmp <- tempfile(fileext = ".txt")
write('Column_1\tColumn_2\tColumn_3
#!{Description}\t\t
#!{Type}E\tE\tE
-1.860574\t-0.3910594\t0.2870352
NaN\t-0.4742951\t0.849998', file=tmp)
mdata <- read.perseus(tmp)
```

`singleChoiceParamInd` *Single choice index*

Description

Extract the index chosen in an `BoolParam`

Usage

```
singleChoiceParamInd(parameters, name)
```

Arguments

`parameters`	The parameters object (see `parseParameters`)
`name`	The name of the parameter

Value

The selected index

Examples

```
tmp <- tempfile(fileext = ".xml")
write('<SingleChoiceParam Name="test_single">\n<Value>1</Value>\n
<Values>\n<Item>A</Item>\n<Item>B</Item>\n</Values>\n</SingleChoiceParam>', file=tmp)
parameters <- parseParameters(tmp)
singleChoiceParamInd(parameters, "test_single")
```

---

singleChoiceParamValue

Single choice value

Description

Extract the value selected in a `SingleChoiceParam`.

Usage

```
singleChoiceParamValue(parameters, name)
```

Arguments

`parameters`	The parameters object (see `parseParameters`)
`name`	The name of the parameter

Value

The string representing the value

Examples

```
tmp <- tempfile(fileext = ".xml")
write('<SingleChoiceParam Name="test_single">\n<Value>1</Value>\n
<Values>\n<Item>A</Item>\n<Item>B</Item>\n</Values>\n</SingleChoiceParam>', file=tmp)
parameters <- parseParameters(tmp)
singleChoiceParamValue(parameters, "test_single")
```

`write.perseus` *write.perseus: function to generate a perseus-readable text document*

Description

Write data to a perseus text file or connection

Write Data to file in the custom Perseus matrix file format.

Usage

```
write.perseus(object = NULL, con = NULL, ...)

## Default S3 method:
write.perseus(object = NULL, con = NULL, main,
  annotCols = NULL, annotRows = NULL, descr = NULL,
  imputeData = NULL, qualityData = NULL, ...)

## S3 method for class 'matrixData'
write.perseus(object, con, ...)

## S3 method for class 'list'
write.perseus(object, con, ...)

## S3 method for class 'data.frame'
write.perseus(object, con, annotCols = NULL, ...)

## S3 method for class 'matrix'
write.perseus(object, con, annotCols = NULL, ...)

## S3 method for class 'ExpressionSet'
write.perseus(object, con, ...)
```

Arguments

`object`	an expressionSet, matrixData, list or table-like object.
`con`	A `connection` object or the path to output file
`...`	additional arguments passed to other functions
`main`	a data frame containing
`annotCols`	a df containing columns containing metadata (about the rows)
`annotRows`	a df containing columns containing metadata (about the columns)
`descr`	a character vector that describes the columns in main and in annotCols (in that order)
`imputeData`	a df containing imputations – True or False of main data frame
`qualityData`	a df containing quality values of main data frame

Value

writes to disk a perseus-interpretable text representation of an R object

NULL

NULL

NULL

NULL

NULL

See Also

read.perseus matrixData

Examples

```
df <- matrixData(
main=data.frame(a=1:3, b=6:8),
annotCols=data.frame(b=c('a','b','c')),
annotRows=data.frame(x=factor(c('1','1'))),
description=c('a','a','b'))
con <- textConnection('df1', 'w')
write.perseus(df, con)
close(con)
```

Index

∗ **matrixData basic functions**
annotCols, 2
annotCols<-, 3
annotRows, 4
annotRows<-, 4
description, 7
description<-, 8
imputeData, 9
imputeData<-, 9
main, 12
main<-, 12
matrixData, 13
matrixData-class, 14
names,matrixData-method, 15
qualityData, 17
qualityData<-, 17

annotCols, 2, *3–5*, *7–10*, *12–14*, *16–18*
annotCols<-, 3
annotRows, *3*, 4, *5*, *7–10*, *12–14*, *16–18*
annotRows<-, 4
as.ExpressionSet.matrixData, 5
as.matrixData.ExpressionSet, 6

boolParamValue, 6

connection, *18, 19*, *21*
create_annotRows, 7

description, *3–5*, 7, *8–10*, *12–14*, *16–18*
description<-, 8

ExpressionSet, *5, 6*

imputeData, *3–5*, *7, 8*, 9, *10*, *12–14*, *16–18*
imputeData<-, 9
infer_perseus_annotation_types, 10
initialize,matrixData-method, 11
intParamValue, 11
isSeekable, *19*

main, *3–5*, *7–10*, 12, *13, 14*, *16–18*
main<-, 12
mapvalues, *10*
matrixData, *3–10*, *12, 13*, 13, *14*, *16–19*, *22*
matrixData-class, 14
MatrixDataCheck, 14

names,matrixData-method, 15
names.matrixData, 16

parseParameters, *6*, *11*, 16, *19, 20*

qualityData, *3–5*, *7–10*, *12–14*, *16*, 17, *18*
qualityData<-, 17

read.perseus, *7*, *22*
read.perseus *(*read.perseus.default*)*, 18
read.perseus.default, 18

singleChoiceParamInd, 19
singleChoiceParamValue, 20

write.perseus, *19*, 21