

# Package ‘do’

July 22, 2025

**Type** Package

**Title** Data Operator

**Version** 2.0.0.1

**Description** Flexibly convert data between long and wide format using just two functions: `reshape_toLong()` and `reshape_toWide()`.

**License** GPL-3

**Encoding** UTF-8

**Imports** data.table, plyr, tmcn, methods, tidymodels, reshape2, tidyr, rvest, xml2, crayon, httr, usethis, desc, utils, rstudioapi, stringr, openxlsx

**RoxygenNote** 7.1.1

**URL** <https://github.com/yikeshu0611/do>

**BugReports** <https://github.com/yikeshu0611/do/issues>

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**Depends** R (>= 4.1)

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2024-09-11 17:20:37 UTC

**Author** Jing Zhang [aut, cre],  
Zhi Jin [aut],  
CRAN Team [ctb]

**Maintainer** Jing Zhang <zj391120@163.com>

## Contents

|                          |   |
|--------------------------|---|
| add_biocViews . . . . .  | 4 |
| all_children . . . . .   | 4 |
| Apriori.Basket . . . . . | 5 |

|                            |    |
|----------------------------|----|
| as.data.frame . . . . .    | 6  |
| as.transactions . . . . .  | 6  |
| attr_href . . . . .        | 7  |
| c.xml_nodeset . . . . .    | 7  |
| cat_n . . . . .            | 8  |
| character.nms . . . . .    | 8  |
| chinese_utf8 . . . . .     | 9  |
| cnOS . . . . .             | 9  |
| columntrans . . . . .      | 10 |
| col_split . . . . .        | 10 |
| compare . . . . .          | 11 |
| complete.data . . . . .    | 12 |
| current_mirror . . . . .   | 12 |
| decrease . . . . .         | 13 |
| delete_left . . . . .      | 13 |
| delete_up . . . . .        | 14 |
| deparse0 . . . . .         | 15 |
| desc2df . . . . .          | 15 |
| dump.it . . . . .          | 16 |
| dup.connect . . . . .      | 16 |
| uplicated_all . . . . .    | 17 |
| uplicated_last . . . . .   | 18 |
| equal_length . . . . .     | 18 |
| exec . . . . .             | 19 |
| expand . . . . .           | 20 |
| factor.nms . . . . .       | 20 |
| file.dir . . . . .         | 21 |
| file.name . . . . .        | 21 |
| fmt . . . . .              | 22 |
| formal_dir . . . . .       | 23 |
| getBiocmirrors . . . . .   | 23 |
| get_names . . . . .        | 24 |
| give_names . . . . .       | 24 |
| Grepl . . . . .            | 25 |
| has_children . . . . .     | 26 |
| in1 . . . . .              | 27 |
| increase . . . . .         | 27 |
| inner_Add_Symbol . . . . . | 28 |
| insertglue . . . . .       | 28 |
| install_Rversion . . . . . | 29 |
| is.dir . . . . .           | 29 |
| is.linux . . . . .         | 30 |
| is.mac . . . . .           | 30 |
| is.windows . . . . .       | 31 |
| join . . . . .             | 31 |
| keep . . . . .             | 32 |
| knife . . . . .            | 33 |
| last . . . . .             | 33 |

|                           |    |
|---------------------------|----|
| last_column . . . . .     | 34 |
| last_row . . . . .        | 34 |
| left . . . . .            | 35 |
| left_equal . . . . .      | 35 |
| legal . . . . .           | 36 |
| list1 . . . . .           | 37 |
| load_extdata . . . . .    | 37 |
| mid . . . . .             | 38 |
| mirror.speed . . . . .    | 38 |
| model.data . . . . .      | 39 |
| NA.col.prob . . . . .     | 40 |
| NA.col.sums . . . . .     | 40 |
| NA.row.prob . . . . .     | 41 |
| NA.row.sums . . . . .     | 41 |
| NA.whole.prob . . . . .   | 42 |
| NA.whole.sums . . . . .   | 42 |
| names_n . . . . .         | 43 |
| Nchar . . . . .           | 43 |
| numeric.nms . . . . .     | 44 |
| package_all . . . . .     | 44 |
| paste0_columns . . . . .  | 45 |
| pipe . . . . .            | 45 |
| rd2df . . . . .           | 46 |
| read_R . . . . .          | 46 |
| Replace . . . . .         | 47 |
| Replace0 . . . . .        | 48 |
| Replace_ex . . . . .      | 48 |
| replicate . . . . .       | 49 |
| reshape_toLong . . . . .  | 50 |
| reshape_toWide . . . . .  | 50 |
| reverse . . . . .         | 51 |
| right . . . . .           | 52 |
| right_equal . . . . .     | 52 |
| rm_all . . . . .          | 53 |
| rm_nchar . . . . .        | 53 |
| row.freq . . . . .        | 54 |
| select . . . . .          | 54 |
| seq_range . . . . .       | 55 |
| set_Bioc_mirror . . . . . | 56 |
| set_CRAN_mirror . . . . . | 57 |
| show_function . . . . .   | 57 |
| split_expand . . . . .    | 58 |
| table_NA . . . . .        | 58 |
| take_out . . . . .        | 59 |
| Trim . . . . .            | 59 |
| unique_no.NA . . . . .    | 60 |
| unlibray . . . . .        | 60 |
| upper.dir . . . . .       | 61 |

|                      |           |
|----------------------|-----------|
| write_xlsx . . . . . | 61        |
| %==% . . . . .       | 62        |
| %+% . . . . .        | 63        |
| %S=% . . . . .       | 63        |
| <b>Index</b>         | <b>64</b> |

---

|               |                                                |
|---------------|------------------------------------------------|
| add_biocViews | <i>Add biocViews Field to Description File</i> |
|---------------|------------------------------------------------|

---

**Description**

Add biocViews Field to Description File

**Usage**

add\_biocViews(value = "", overwrite = TRUE)

**Arguments**

|           |                            |
|-----------|----------------------------|
| value     | package names              |
| overwrite | logical, TRUE is defaulted |

---

|              |                                   |
|--------------|-----------------------------------|
| all_children | <i>Extract all children nodes</i> |
|--------------|-----------------------------------|

---

**Description**

Extract all children nodes

**Usage**

all\_children(x, res = "do not change", i = 1)

**Arguments**

|     |                                             |
|-----|---------------------------------------------|
| x   | one or more documents, nodes, or node sets. |
| res | omit. do not make any change.               |
| i   | must be 1                                   |

**Value**

nodeset

**Examples**

```
txt='<div class="activityBody postBody thing">
  <p>
    <a href="/forum/conversation" class="mqPostRef">(22)</a>
    where?
  </p>
  <p>
    stays
    <b>disappears</b>
    <a>disappears</a>
    <span>disappears</span>
    stays
  </p>
</div>'
library(xml2)
html=read_html(txt)

all_children(html)
```

---

Apriori.Basket

---

*Convert vector to sparse matrix*


---

**Description**

Convert vector or dataframe to sparse matrix.

**Usage**

```
Apriori.Basket(x, sep = ";", dup.delete = FALSE)
```

**Arguments**

|            |                                                                       |
|------------|-----------------------------------------------------------------------|
| x          | a vector                                                              |
| sep        | one separator                                                         |
| dup.delete | whether to delete duplicated values in the same row, default is FALSE |

**Value**

a sparse matrix

**Examples**

```
# convert a vector to sparse matrix
g=c('a,b,a,,','a,b,c,d','d,c,f,g,h')
Apriori.Basket(x=g,sep = ',')

# convert a dataframe to sparse matrix
library(data.table)
```

```
df=fread(text = '
t1 t2 t3
a NA d
g a j')
Apriori.Basket(x=df,sep = ',')
```

---

|               |                                                                |
|---------------|----------------------------------------------------------------|
| as.data.frame | <i>Transform to dataframe rules object or calibrate object</i> |
|---------------|----------------------------------------------------------------|

---

**Description**

Transform to dataframe rules object or calibrate object

**Usage**

```
## S3 method for class 'rules'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S3 method for class 'calibrate'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

**Arguments**

|           |                                            |
|-----------|--------------------------------------------|
| x         | data with rules class for package 'arules' |
| row.names | ignore                                     |
| optional  | ignore                                     |
| ...       | ignore                                     |

**Value**

a dataframe

---

|                 |                                  |
|-----------------|----------------------------------|
| as.transactions | <i>Transform to transactions</i> |
|-----------------|----------------------------------|

---

**Description**

Transform to transactions

**Usage**

```
as.transactions(x)
```

**Arguments**

|   |                     |
|---|---------------------|
| x | dataframe or matrix |
|---|---------------------|

**Value**

a transaction data

---

|           |                                           |
|-----------|-------------------------------------------|
| attr_href | <i>Get hypertext reference attributes</i> |
|-----------|-------------------------------------------|

---

**Description**

Get hypertext reference attributes

**Usage**

attr\_href(x)

**Arguments**

x                    A document (from read\_html()), node set (from html\_elements()), node (from html\_element()), or session (from session()).

**Value**

hypertext reference attributes

---

|               |                           |
|---------------|---------------------------|
| c.xml_nodeset | <i>Comine xml_nodeset</i> |
|---------------|---------------------------|

---

**Description**

Comine xml\_nodeset

**Usage**

```
## S3 method for class 'xml_nodeset'  
c(...)
```

**Arguments**

...                    one or more xml\_nodeset

**Value**

xml\_nodeset

---

|       |                              |
|-------|------------------------------|
| cat_n | <i>print vector by lines</i> |
|-------|------------------------------|

---

**Description**

print vector by lines

**Usage**

```
cat_n(x, n = 3, ind = 0)
```

**Arguments**

|     |                                              |
|-----|----------------------------------------------|
| x   | one vector                                   |
| n   | number if element in each line, default is 3 |
| ind | indentation, default is 0                    |

**Value**

print vector by lines

**Examples**

```
cat_n(1:10)
cat_n(1:10, ind=3)
```

---

|               |                                                      |
|---------------|------------------------------------------------------|
| character.nms | <i>Return character names in matrix or dataframe</i> |
|---------------|------------------------------------------------------|

---

**Description**

Return character names in matrix or dataframe

**Usage**

```
character.nms(df)
```

**Arguments**

|    |                     |
|----|---------------------|
| df | dataframe or matrix |
|----|---------------------|

**Value**

character names vectors

---

|              |                              |
|--------------|------------------------------|
| chinese_utf8 | <i>UTF8 Code for Chinese</i> |
|--------------|------------------------------|

---

**Description**

UTF8 Code for Chinese

**Usage**

chinese\_utf8(x)

**Arguments**

x                      chinese characters

**Value**

an expression with UTF8 code.

---

|      |                                                                                  |
|------|----------------------------------------------------------------------------------|
| cnOS | <i>Chinese operating system Whether the computer is Chinese operating system</i> |
|------|----------------------------------------------------------------------------------|

---

**Description**

Chinese operating system Whether the computer is Chinese operating system

**Usage**

cnOS()

**Value**

logical

**Examples**

cnOS()

---

|             |                         |
|-------------|-------------------------|
| columntrans | <i>Change data type</i> |
|-------------|-------------------------|

---

**Description**

Change data type

**Usage**

```
factor.it(x) <- value  
  
factor.it(x, value)  
  
numeric.it(x, value)  
  
numeric.it(x) <- value
```

**Arguments**

|       |              |
|-------|--------------|
| x     | dataframe    |
| value | column names |

**Value**

factor or numeric columns in a dataframe

**Examples**

```
str(mtcars)  
factor.it(mtcars,c("cyl", "vs", "am", "gear"))  
factor.it(mtcars)=c("cyl", "vs", "am", "gear")  
str(mtcars)  
  
numeric.it(mtcars,c("cyl", "vs", "am", "gear"))  
numeric.it(mtcars)=c("cyl", "vs", "am", "gear")  
str(mtcars)
```

---

|           |                                    |
|-----------|------------------------------------|
| col_split | <i>Split A Vector into Columns</i> |
|-----------|------------------------------------|

---

**Description**

Split A Vector into Columns

**Usage**

```
col_split(x, split, reg_expr, colnames, cat = TRUE)
```

**Arguments**

|          |                                         |
|----------|-----------------------------------------|
| x        | a vector                                |
| split    | one or more characters. Split exactly   |
| reg_expr | character. Split by regular expressions |
| colnames | optional. Column names for outcome      |
| cat      | logical, whether to show message        |

**Value**

A dataframe with several columns.

**Examples**

```
x=c('1a2','3a4','4a4')
col_split(x,split='a')
col_split(x = x,reg_expr = '[a-z]')

#two splits
df=data.frame(result=c('A, B-C',
                        'A, C-D',
                        'E, F-G'))
col_split(x = df[,1],split = c(',','-'))
```

---

|         |                            |
|---------|----------------------------|
| compare | <i>Compare two vectors</i> |
|---------|----------------------------|

---

**Description**

Compare two vectors

**Usage**

```
equal(a, b)

over(a, b)

lower(a, b)
```

**Arguments**

|   |                  |
|---|------------------|
| a | one vector       |
| b | the other vector |

**Value**

the compared object

Examples

```
equal(letters,c('a','b'))
over(1:10,5)
over(1:10,5)
```

---

|               |                      |
|---------------|----------------------|
| complete.data | <i>Complete data</i> |
|---------------|----------------------|

---

Description

Removing rows with NA in dataframe or matrix. Removing NA atomic.

Usage

```
complete.data(x)
```

Arguments

x                      dataframe or matrix or atomic

Value

complete data

Examples

```
x=c(1,NA,2)
complete.data(x)

x=data.frame(a=c(1,NA))
complete.data(x)
```

---

|                |                       |
|----------------|-----------------------|
| current_mirror | <i>Current mirror</i> |
|----------------|-----------------------|

---

Description

Current mirrors of CRAN adn Bioconductor

Usage

```
current_mirror()
```

Value

a list contains CRAN and Bioconductor mirror

Examples

```
current_mirror()
```

---

|          |                           |
|----------|---------------------------|
| decrease | <i>decrease character</i> |
|----------|---------------------------|

---

Description

decrease character

Usage

```
decrease(chr)
```

Arguments

chr                    one character vector

Value

decreased vector

Examples

```
set.seed(2020)
x=rnorm(20)
decrease(x)
```

---

|             |                                             |
|-------------|---------------------------------------------|
| delete_left | <i>Delete and Move Left the rest Values</i> |
|-------------|---------------------------------------------|

---

Description

Delete and Move Left the rest Values

Usage

```
delete_left(x, delete)
```

Arguments

x                    dataframe or matrix  
delete                one delete object

Value

dataframe or matrix

Examples

```
a=c(1,NA,7,NA)
b=c(NA,2,2,7)
d=c(1,NA,40,7)
df=data.frame(a,b,d)
delete_left(x=df,NA)
```

---

|           |                                    |
|-----------|------------------------------------|
| delete_up | Delete and Move Up the Rest Values |
|-----------|------------------------------------|

---

Description

Delete and Move Up the Rest Values

Usage

```
delete_up(x, delete)
```

Arguments

- x                      dataframe or matrix
- delete                one delete object

Value

dataframe or matrix

Examples

```
a=c(1,NA,7,NA)
b=c(NA,2,2,7)
d=c(1,NA,40,7)
df=data.frame(a,b,d)

delete_up(x = df,delete = NA)
```

---

|          |                                       |
|----------|---------------------------------------|
| deparse0 | <i>substitutue, deparse and paste</i> |
|----------|---------------------------------------|

---

**Description**

substitutue, deparse and paste

**Usage**

```
deparse0(x)
```

**Arguments**

x                      one object

**Value**

character

**Examples**

```
deparse0(j)
```

---

|         |                                                      |
|---------|------------------------------------------------------|
| desc2df | <i>Convert package description file to dataframe</i> |
|---------|------------------------------------------------------|

---

**Description**

Convert package description file to dataframe

**Usage**

```
desc2df(desc)
```

**Arguments**

desc                      description file path

**Value**

One dataframe with column names of field

---

|         |                                 |
|---------|---------------------------------|
| dump.it | Create dump matrix for a vector |
|---------|---------------------------------|

---

**Description**

Create dump matrix for a vector

**Usage**

```
dump.it(..., include.name = TRUE)
```

**Arguments**

|              |                                                              |
|--------------|--------------------------------------------------------------|
| ...          | one vector                                                   |
| include.name | logical, default is TRUE, wether to include name of variable |

**Value**

a dump matix contains 0 and 1

**Examples**

```
x=c('a','b','c','a','a')
dump.it(x)
dump.it(mtcars$am)
dump.it(mtcars[, 'am'])
```

---

|             |                           |
|-------------|---------------------------|
| dup.connect | Connect Duplicated Values |
|-------------|---------------------------|

---

**Description**

Connect Duplicated Values

**Usage**

```
dup.connect(data, id, dup.var)
```

**Arguments**

|         |                                   |
|---------|-----------------------------------|
| data    | dataframe or matrix               |
| id      | id column names or indexs         |
| dup.var | duplicated column names or indexs |

**Value**

dataframe contains id and duplicated values

**Examples**

```
dup.connect(data = mtcars, id = 'am', dup.var = 'cyl')
dup.connect(data = mtcars,
            id = c('am', 'gear'),
            dup.var = c('cyl', 'qsec'))
```

---

|                |                                         |
|----------------|-----------------------------------------|
| duplicated_all | <i>Determine All Duplicate Elements</i> |
|----------------|-----------------------------------------|

---

**Description**

Determine All Duplicate Elements

**Usage**

```
duplicated_all(x)
```

**Arguments**

|   |           |
|---|-----------|
| x | character |
|---|-----------|

**Value**

logical value

**Examples**

```
x=c(1,3,2,1,2)
duplicated(x)
duplicated_all(x)
```

---

|                 |                                                          |
|-----------------|----------------------------------------------------------|
| duplicated_last | <i>Determine Duplicate Elements in the Last Position</i> |
|-----------------|----------------------------------------------------------|

---

**Description**

Determine Duplicate Elements in the Last Position

**Usage**

duplicated\_last(x)

**Arguments**

x                      character

**Value**

logical value

**Examples**

```
x=c(1,3,2,1,2)
duplicated(x)
duplicated_last(x)
```

---

|              |                     |
|--------------|---------------------|
| equal_length | <i>Equal Length</i> |
|--------------|---------------------|

---

**Description**

Equal Length

**Usage**

equal\_length(x, suffix = " ", nchar, colname = FALSE, rowname = FALSE)

**Arguments**

x                      can be number, strings, verctors, dataframe or matrix.  
suffix                suffix  
nchar                maximun length  
colname              a logistic value, default is FALSE  
rowname              a logistic value, default is FALSE

**Value**

equal length results

**Examples**

```
a=c(123,1,24,5,1.22554)
equal_length(a,0)

df = data.frame(
  a=c(12,1,1.23),
  b=c('a','abcd','d')
)
equal_length(x = df,suffix = 'x')

equal_length(x = df,suffix = 0,nchar =5)
```

---

|      |                                                                              |
|------|------------------------------------------------------------------------------|
| exec | <i>execute string command This command just execute in the parent frame.</i> |
|------|------------------------------------------------------------------------------|

---

**Description**

execute string command This command just execute in the parent frame.

**Usage**

```
exec(string, envir = parent.frame())
```

**Arguments**

|        |                                                    |
|--------|----------------------------------------------------|
| string | one string                                         |
| envir  | the environment in which sting is to be evaluated. |

**Value**

execute string command

**Examples**

```
a=2
exec('a = 1')
a
```

---

|        |                              |
|--------|------------------------------|
| expand | <i>Expand Data by Weight</i> |
|--------|------------------------------|

---

**Description**

Expand Data by Weight

**Usage**

```
expand(x, weight)
```

**Arguments**

|        |                              |
|--------|------------------------------|
| x      | dataframe or matrix          |
| weight | weight column names or index |

**Value**

expanded data

**Examples**

```
df=data.frame(v=c(1,2,3),
              x=c(7,8,9),
              n=c(2,3,4))
expand(x = df,weight = 3)
expand(x = df,weight = 'n')
```

---

|            |                                                   |
|------------|---------------------------------------------------|
| factor.nms | <i>Return factor names in matrix or dataframe</i> |
|------------|---------------------------------------------------|

---

**Description**

Return factor names in matrix or dataframe

**Usage**

```
factor.nms(df)
```

**Arguments**

|    |                     |
|----|---------------------|
| df | dataframe or matrix |
|----|---------------------|

**Value**

factor names vectors

---

|          |                           |
|----------|---------------------------|
| file.dir | <i>up level directory</i> |
|----------|---------------------------|

---

**Description**

up level directory

**Usage**

```
file.dir(path, end.slash = TRUE, extension = TRUE)
```

**Arguments**

|           |                                              |
|-----------|----------------------------------------------|
| path      | path of file                                 |
| end.slash | logical. Whether to end with slash           |
| extension | logical. whether file name include extension |

**Value**

upper directory

---

|           |                          |
|-----------|--------------------------|
| file.name | <i>Extract file name</i> |
|-----------|--------------------------|

---

**Description**

Extract file name

**Usage**

```
file.name(..., extension = TRUE)
```

**Arguments**

|           |                                            |
|-----------|--------------------------------------------|
| ...       | one or more file path                      |
| extension | whether include extension, default is TRUE |

**Value**

file names

**Examples**

```
file.name('f:/dir/1.txt')
file.name('f:/dir/1.txt', 'f:/dir/1.txt')
file.name('f:/dir/1.txt', 'f:/dir/1.txt', 'f:/dir/')

```

fmt

*Formatting Replacement***Description**

Formatting Replacement

**Usage**

fmt(x, ...)

**Arguments**

x                    format with slash number and one space, which is like "/1 ". Number means replacement order.

...                  values to be passed into x

**Value**

replaced string

**Examples**

```
'whwdzg, ykybnfg'
fmt('/ hwdzg, ykybnfg',
    'w')
fmt('/ h/ dzg, ykybnfg',
    'w')
fmt('/1 h/ dzg, ykybnfg',
    'w')
fmt('/1 h/ dzg, ykybnfg',
    'w', '-w-')

fmt('/ h/1 dzg, ykybnfg',
    'w', '-w-')

fmt('/1 h/0 dzg, ykybnfg',
    'w', '-w-')

'|w|' |>
  fmt(x = '/ h/ dzg, ykybnfg')

'|w|' |>
  fmt(x = '/ h/ dzg, ykybnfg',
      '-w-')

'|w|' |>
  fmt(x = '/ h/1 dzg, ykybnfg',
      '-w-')
```

---

|            |                         |
|------------|-------------------------|
| formal_dir | <i>formal directory</i> |
|------------|-------------------------|

---

**Description**

formal directory

**Usage**

```
formal_dir(dir, end.slash = FALSE)
```

**Arguments**

|           |               |
|-----------|---------------|
| dir       | one directory |
| end.slash | logical       |

**Value**

formed directory

---

|                |                                 |
|----------------|---------------------------------|
| getBiocmirrors | <i>get bioconductor mirrors</i> |
|----------------|---------------------------------|

---

**Description**

get bioconductor mirrors

**Usage**

```
getBiocmirrors()
```

**Value**

bioconductor mirrors

---

get\_names

*Get Names of Object*


---

**Description**

Return the names of input. For example: if you input a, you will get 'a'.

**Usage**

```
get_names(...)
```

**Arguments**

... any type of data object

**Value**

names of object

**Examples**

```
a=c(1,2,3)
get_names(a,mtcars)
```

---

give\_names

*change vector, dataframe or matrix names*


---

**Description**

change vector, dataframe or matrix names

**Usage**

```
give_names(data, ...)

## S3 method for class 'character'
give_names(data, ...)

## S3 method for class 'numeric'
give_names(data, ...)

## S3 method for class 'logical'
give_names(data, ...)

## S3 method for class 'list'
give_names(data, ...)
```

```
## S3 method for class 'data.frame'
give_names(data, ..., row = FALSE)

## S3 method for class 'matrix'
give_names(data, ..., row = FALSE)
```

**Arguments**

|      |                                                           |
|------|-----------------------------------------------------------|
| data | one vector, list, dataframe or matrix                     |
| ...  | one or more names                                         |
| row  | logical, whether the names is row names. Default is FALSE |

**Value**

names changed data

---

|       |                                     |
|-------|-------------------------------------|
| Grepl | <i>Judge for Included Character</i> |
|-------|-------------------------------------|

---

**Description**

Judge for Included Character

**Usage**

Grepl(pattern, x)

**Arguments**

|         |                     |
|---------|---------------------|
| pattern | one or more vectors |
| x       | one or more vectors |

**Details**

,

**Value**

a matrix with logical words

**Examples**

```

a=c('abcd','agj','abcu')

# Grepl for one vector
pat1='b'
Grepl(pat1,a)

# Grepl for two vectors
pat2=c('c','d')
Grepl(pat2,a)

# use %or% in pattern
pat3=c('a%or%c','d')
Grepl(pat3,a)

# use %and% in pattern
pat4=c('a%and%c','d')
Grepl(pat4,a)

```

---

|              |                                    |
|--------------|------------------------------------|
| has_children | <i>Wether children nodes exist</i> |
|--------------|------------------------------------|

---

**Description**

Wether children nodes exist

**Usage**

```
has_children(...)
```

**Arguments**

... one or more documents, nodes, or node sets.

**Value**

logical value

**Examples**

```

txt='<div class="activityBody postBody thing">
  <p>
    <a href="/forum/conversation" rel="post" >(22)</a>
    where?
  </p>
  <p>
    stays
    <b>disappears</b>
    <a>disappears</a>D
  </p>
</div>'

```

```
      <span>disappears</span>
      stays
    </p>
  </div>'
library(xml2)
html=read_html(txt)
has_children(html)
```

---

|     |           |
|-----|-----------|
| in1 | <i>in</i> |
|-----|-----------|

---

**Description**

in

**Usage**

in1()

---

|          |                           |
|----------|---------------------------|
| increase | <i>increase character</i> |
|----------|---------------------------|

---

**Description**

increase character

**Usage**

increase(chr)

**Arguments**

chr                    one vector

**Value**

increased vector

**Examples**

```
set.seed(2020)
x=rnorm(20)
increase(x)
```

---

|                               |                            |
|-------------------------------|----------------------------|
| <code>inner_Add_Symbol</code> | <i>Concatenate Strings</i> |
|-------------------------------|----------------------------|

---

**Description**

Concatenate vectors by adding a symbol.

**Usage**

```
inner_Add_Symbol(x, symbol = "+")
```

**Arguments**

|                     |                |
|---------------------|----------------|
| <code>x</code>      | vectors        |
| <code>symbol</code> | defulat is '+' |

**Value**

a concatenated string

**Examples**

```
inner_Add_Symbol(c('a','b'))
inner_Add_Symbol(c('a','b'),"$")
inner_Add_Symbol(c('a','b'),"")
```

---

|                         |             |
|-------------------------|-------------|
| <code>insertglue</code> | <i>glue</i> |
|-------------------------|-------------|

---

**Description**

glue

**Usage**

```
insertglue()
```

---

|                  |                                                  |
|------------------|--------------------------------------------------|
| install_Rversion | <i>Install contributed packages by R version</i> |
|------------------|--------------------------------------------------|

---

**Description**

Install contributed packages by R version

**Usage**

```
install_Rversion(..., platform, Rversion = NULL, lib = ".")
```

**Arguments**

|          |                     |
|----------|---------------------|
| ...      | one or more package |
| platform | windows or mac      |
| Rversion | version of R        |
| lib      | path                |

**Value**

contributed packages

---

|        |                                       |
|--------|---------------------------------------|
| is.dir | <i>Whether file path is directory</i> |
|--------|---------------------------------------|

---

**Description**

Whether file path is directory

**Usage**

```
is.dir(...)
```

**Arguments**

|     |                       |
|-----|-----------------------|
| ... | one or more file path |
|-----|-----------------------|

**Value**

logical

---

|          |                         |
|----------|-------------------------|
| is.linux | <i>operation system</i> |
|----------|-------------------------|

---

**Description**

operation system

**Usage**

is.linux()

**Value**

logical

**Examples**

is.linux()

---

|        |                         |
|--------|-------------------------|
| is.mac | <i>operation system</i> |
|--------|-------------------------|

---

**Description**

operation system

**Usage**

is.mac()

**Value**

logical

**Examples**

is.mac()

---

|            |                         |
|------------|-------------------------|
| is.windows | <i>operation system</i> |
|------------|-------------------------|

---

**Description**

operation system

**Usage**

```
is.windows()
```

**Value**

logical

**Examples**

```
is.windows()
```

---

|      |                                     |
|------|-------------------------------------|
| join | <i>Join two dataframes together</i> |
|------|-------------------------------------|

---

**Description**

Join two dataframes by the same id column.

**Usage**

```
join_inner(x, y, by = NULL)
```

```
join_full(x, y, by = NULL)
```

```
join_left(x, y, by = NULL)
```

```
join_right(x, y, by = NULL)
```

```
join_out(x, y, by = NULL)
```

**Arguments**

|    |                                  |
|----|----------------------------------|
| x  | one dataframe                    |
| y  | the other dataframe              |
| by | the id name in x and y dataframe |

**Details**

join\_inner(), join\_full(), join\_left(), join\_right() and join\_out() are five functons to joint two dataframes together. They are based on package 'data.table', so they are more efficient and fast.

**Value**

one joined dataframe.

**Examples**

```
df1=data.frame(x=rep(c('b','a','c'),each=3),
               y=c(1,3,6),
               v=1:9)

df2=data.frame(x=c('c','b','e'),
               v=8:6,
               foo=c(4,2,1))
join_inner(df1,df2,'x')
join_full(df1,df2,'x')
join_left(df1,df2,'x')
join_right(df1,df2,'x')
join_out(df1,df2,'x')
```

---

|      |                     |
|------|---------------------|
| keep | <i>Keep objects</i> |
|------|---------------------|

---

**Description**

Keep objects

**Usage**

```
keep(..., envir = .GlobalEnv)
```

**Arguments**

- ... one or more objects
- envir environment, default is global

**Examples**

```
a <- 1
b <- 2
d <- 4
keep(a)
```

---

|       |                         |
|-------|-------------------------|
| knife | <i>Knife characters</i> |
|-------|-------------------------|

---

**Description**

Knife characters

**Usage**

```
knife_left(x, n)
```

```
knife_right(x, n)
```

**Arguments**

|   |               |
|---|---------------|
| x | one character |
| n | number        |

**Examples**

```
knife_left(123,2)  
knife_right(123,2)
```

---

|      |                                   |
|------|-----------------------------------|
| last | <i>Select character from last</i> |
|------|-----------------------------------|

---

**Description**

Select character from last

**Usage**

```
last(x, n)
```

**Arguments**

|   |                                            |
|---|--------------------------------------------|
| x | vector                                     |
| n | If missing, the last element will be used. |

**Value**

last element

**Examples**

```
letters |> last()  
letters |> last(1:2)
```

---

|             |                                   |
|-------------|-----------------------------------|
| last_column | Select dataframe column from last |
|-------------|-----------------------------------|

---

**Description**

Select dataframe column from last

**Usage**

```
last_column(x, n)
```

**Arguments**

|   |                                            |
|---|--------------------------------------------|
| x | dataframe                                  |
| n | If missing, the last element will be used. |

**Value**

last column

**Examples**

```
mtcars |> last_column()  
mtcars |> last_column(1:2)
```

---

|          |                                |
|----------|--------------------------------|
| last_row | Select dataframe row from last |
|----------|--------------------------------|

---

**Description**

Select dataframe row from last

**Usage**

```
last_row(x, n)
```

**Arguments**

|   |                                            |
|---|--------------------------------------------|
| x | dataframe                                  |
| n | If missing, the last element will be used. |

**Value**

last row

**Examples**

```
mtcars |> last_row()
mtcars |> last_row(1:2)
```

---

|      |                                          |
|------|------------------------------------------|
| left | <i>Truncate Characters from the Left</i> |
|------|------------------------------------------|

---

**Description**

Truncate Characters from the Left

**Usage**

```
left(x, n)
```

**Arguments**

|   |                                                       |
|---|-------------------------------------------------------|
| x | can be number, strings, vectors, dataframe or matrix. |
| n | length                                                |

**Value**

substring

**Examples**

```
left("abcd",3)
left(c("abc","gjh"),2)
df = data.frame(
  a = c(123,234,456),
  b = c("abc","bcd","hjk")
)
left(df,2)
```

---

|            |                                                                                           |
|------------|-------------------------------------------------------------------------------------------|
| left_equal | <i>Compare two characters from left Much useful for arguments input. Case is ignored.</i> |
|------------|-------------------------------------------------------------------------------------------|

---

**Description**

Compare two characters from left Much useful for arguments input. Case is ignored.

**Usage**

```
left_equal(x1, x2)
```

**Arguments**

|    |                     |
|----|---------------------|
| x1 | one character       |
| x2 | the other character |

**Value**

logical

**Examples**

```
left_equal('o', 'OK')  
left_equal('ok', 'O')  
left_equal('ok', 'Ok')
```

---

|       |                                                                                               |
|-------|-----------------------------------------------------------------------------------------------|
| legal | <i>Check legal character Whether the character is legal for names in dataframe or formula</i> |
|-------|-----------------------------------------------------------------------------------------------|

---

**Description**

Check legal character Whether the character is legal for names in dataframe or formula

**Usage**

```
legal(...)
```

**Arguments**

|     |                    |
|-----|--------------------|
| ... | one or more string |
|-----|--------------------|

**Value**

logical, TRUE means legal.

**Examples**

```
legal('a', 'b', 'a b')
```

---

|       |                 |
|-------|-----------------|
| list1 | Select list one |
|-------|-----------------|

---

**Description**

Select list one

**Usage**

```
list1(x)
```

**Arguments**

|   |      |
|---|------|
| x | list |
|---|------|

**Value**

element in list 1

**Examples**

```
x = list(mtcars)
x |> list1()
```

---

|              |                                   |
|--------------|-----------------------------------|
| load_extdata | Load external data from R package |
|--------------|-----------------------------------|

---

**Description**

Load external data from R package

**Usage**

```
load_extdata(package, file)
```

**Arguments**

|         |                  |
|---------|------------------|
| package | one package name |
| file    | one file name    |

**Value**

path of data

---

|     |                                            |
|-----|--------------------------------------------|
| mid | <i>Truncate Characters from the Inside</i> |
|-----|--------------------------------------------|

---

**Description**

Truncate Characters from the Inside

**Usage**

```
mid(x, start, n = 1e+11)
```

**Arguments**

|       |                                                       |
|-------|-------------------------------------------------------|
| x     | can be number, strings, vectors, dataframe or matrix. |
| start | starting position                                     |
| n     | length, n can be less than zero                       |

**Value**

substring

**Examples**

```
mid("abcd",3,1)
mid(c("abc","gjh"),2,2)
df = data.frame(
  a = c(123,234,456),
  b = c("abc","bcd","hjk")
)
mid(df,2,1)
mid(df,2,-2)
```

---

|              |                             |
|--------------|-----------------------------|
| mirror.speed | <i>Test speed of mirror</i> |
|--------------|-----------------------------|

---

**Description**

Test speed of mirror

**Usage**

```
mirror.speed(min.second = 0.2, cran = TRUE, bioc = TRUE)
```

**Arguments**

|            |                                                                |
|------------|----------------------------------------------------------------|
| min.second | the minium second time to visit the mirror web page            |
| cran       | logical, whether to test CRAN mirrors. Default is TRUE         |
| bioc       | logical, whether to test bioconductor mirrors. Default is TRUE |

**Value**

repositories which visiting time is minus the minium second.

---

|            |                              |
|------------|------------------------------|
| model.data | <i>Extract data of model</i> |
|------------|------------------------------|

---

**Description**

Extract data of model

**Usage**

```
model.data(fit)

model.y(fit)

model.x(fit)
```

**Arguments**

|     |                |
|-----|----------------|
| fit | fitted results |
|-----|----------------|

**Value**

dataframe in the model

**Examples**

```
fit <- lm(mpg~vs+am+poly(qsec,2),data=mtcars)
head(model.data(fit))
model.y(fit)
model.x(fit)
```

---

|             |                                              |
|-------------|----------------------------------------------|
| NA.col.prob | <i>Proportion of missing value by column</i> |
|-------------|----------------------------------------------|

---

**Description**

NA is treated as missing value.

**Usage**

```
NA.col.prob(data)
```

**Arguments**

|      |                             |
|------|-----------------------------|
| data | must be dataframe or matrix |
|------|-----------------------------|

**Value**

proportion of missing value by column

**Examples**

```
df = data.frame(x=rep(c(1,NA,2,NA,6,NA),10),  
                y=rep(c(1,NA,2),20))  
NA.col.prob(df)
```

---

|             |                                       |
|-------------|---------------------------------------|
| NA.col.sums | <i>Sum of missing value by column</i> |
|-------------|---------------------------------------|

---

**Description**

NA is treated as missing value.

**Usage**

```
NA.col.sums(data)
```

**Arguments**

|      |                             |
|------|-----------------------------|
| data | must be dataframe or matrix |
|------|-----------------------------|

**Value**

sum of missing value by column

**Examples**

```
df = data.frame(x=rep(c(1,NA,2,NA,6,NA),10),  
                y=rep(c(1,NA,2),20))  
NA.col.sums(df)
```

---

|             |                                           |
|-------------|-------------------------------------------|
| NA.row.prob | <i>Proportion of missing value by row</i> |
|-------------|-------------------------------------------|

---

**Description**

NA is treated as missing value.

**Usage**

```
NA.row.prob(data)
```

**Arguments**

|      |                             |
|------|-----------------------------|
| data | must be dataframe or matrix |
|------|-----------------------------|

**Value**

proportion of missing value by row

**Examples**

```
df = data.frame(x=rep(c(1,NA,2,NA,6,NA),10),  
                y=rep(c(1,NA,2),20))  
NA.row.prob(df)
```

---

|             |                                    |
|-------------|------------------------------------|
| NA.row.sums | <i>Sum of missing value by row</i> |
|-------------|------------------------------------|

---

**Description**

NA is treated as missing value.

**Usage**

```
NA.row.sums(data)
```

**Arguments**

|      |                             |
|------|-----------------------------|
| data | must be dataframe or matrix |
|------|-----------------------------|

**Value**

sum of missing value by row

**Examples**

```
df = data.frame(x=rep(c(1,NA,2,NA,6,NA),10),  
                y=rep(c(1,NA,2),20))  
NA.row.sums(df)
```

---

|               |                                                           |
|---------------|-----------------------------------------------------------|
| NA.whole.prob | <i>Proportion of missing value in the whole dataframe</i> |
|---------------|-----------------------------------------------------------|

---

**Description**

NA is treated as missing value.

**Usage**

```
NA.whole.prob(data)
```

**Arguments**

|      |                             |
|------|-----------------------------|
| data | must be dataframe or matrix |
|------|-----------------------------|

**Value**

proportion of missing value in the whole dataframe

**Examples**

```
df = data.frame(x=rep(c(1,NA,2,NA,6,NA),10),  
                y=rep(c(1,NA,2),20))  
NA.whole.prob(df)
```

---

|               |                                                    |
|---------------|----------------------------------------------------|
| NA.whole.sums | <i>Sum of missing value in the whole dataframe</i> |
|---------------|----------------------------------------------------|

---

**Description**

NA is treated as missing value.

**Usage**

```
NA.whole.sums(data)
```

**Arguments**

|      |                             |
|------|-----------------------------|
| data | must be dataframe or matrix |
|------|-----------------------------|

**Value**

sum of missing value in the whole dataframe

**Examples**

```
df = data.frame(x=rep(c(1,NA,2,NA,6,NA),10),  
                y=rep(c(1,NA,2),20))  
NA.whole.sums(df)
```

---

|         |                                     |
|---------|-------------------------------------|
| names_n | <i>Names with different letters</i> |
|---------|-------------------------------------|

---

**Description**

Names with different letters

**Usage**

```
names_n(df, most = NULL, least = NULL)
```

**Arguments**

|       |                                                           |
|-------|-----------------------------------------------------------|
| df    | dataframe or matrix                                       |
| most  | names with at most different letters, which means $\leq$  |
| least | names with at least different letters, which means $\geq$ |

**Value**

names

---

|       |                             |
|-------|-----------------------------|
| Nchar | <i>Number of Characters</i> |
|-------|-----------------------------|

---

**Description**

Number of Characters

**Usage**

```
Nchar(x)
```

**Arguments**

|   |                                                       |
|---|-------------------------------------------------------|
| x | can be number, strings, vectors, dataframe or matrix. |
|---|-------------------------------------------------------|

**Value**

number of characters in each location

**Examples**

```
Nchar("abcd")
Nchar(c("abc", "gjh"))
df = data.frame(
  a = c(1,12,12.3),
  b = c("a", "ab", "abc")
)
Nchar(df)
```

---

|             |                                                    |
|-------------|----------------------------------------------------|
| numeric.nms | <i>Return numeric names in matrix or dataframe</i> |
|-------------|----------------------------------------------------|

---

**Description**

Return numeric names in matrix or dataframe

**Usage**

```
numeric.nms(df)
```

**Arguments**

df                      dataframe or matrix

**Value**

numeric names vectors

---

|             |                                         |
|-------------|-----------------------------------------|
| pakcage_all | <i>Get all functions in one package</i> |
|-------------|-----------------------------------------|

---

**Description**

Get all functions in one package

**Usage**

```
pakcage_all(x)
```

**Arguments**

x                      package

**Value**

all functions in one package

---

|                |                               |
|----------------|-------------------------------|
| paste0_columns | <i>Paste Columns Together</i> |
|----------------|-------------------------------|

---

**Description**

Paste each column in a dataframe together.

**Usage**

```
paste0_columns(df, collapse = ",")
```

**Arguments**

df                    a dataframe  
collapse            collapse, default is comma

**Value**

a character

**Examples**

```
df=data.frame(a=c(1,2,30),  
              b=c('x','y','z'))  
paste0_columns(df)  
  
df=data.frame(a=c(1,2,30),b=c('x','y','z'),c=c(1,7,8))  
paste0_columns(df)
```

---

|      |             |
|------|-------------|
| pipe | <i>pipe</i> |
|------|-------------|

---

**Description**

pipe

**Usage**

```
pipe()
```

---

|       |                                                                   |
|-------|-------------------------------------------------------------------|
| rd2df | <i>Convert package Rd file under man directory into dataframe</i> |
|-------|-------------------------------------------------------------------|

---

**Description**

Convert package Rd file under man directory into dataframe

**Usage**

```
rd2df(pkg)
```

**Arguments**

|     |                                              |
|-----|----------------------------------------------|
| pkg | source package path unzip from "tar.gz" file |
|-----|----------------------------------------------|

**Value**

one dataframe

---

|        |                    |
|--------|--------------------|
| read_R | <i>Read R file</i> |
|--------|--------------------|

---

**Description**

Read R file

**Usage**

```
read_R(R, pattern)
```

**Arguments**

|         |                |
|---------|----------------|
| R       | path of R file |
| pattern | pattern        |

**Value**

one vector of R command with names of R file

---

|         |                |
|---------|----------------|
| Replace | <i>Replace</i> |
|---------|----------------|

---

**Description**

There are two methods in this function. You can use repalce many objects to one by form and to. pattern can be used to one object replaced by the other one.

**Usage**

```
Replace(data, from, to, pattern, ignore.case = FALSE)
```

**Arguments**

- data            can be number, strings, verctors, dataframe or matrix.
- from           replaced stings
- to             replacements
- pattern        like from:to
- ignore.case    logical, whether to ignore case

**Value**

replaced data

**Examples**

```
Replace(data = 232,from = 2,to = 1)
Replace(data = c(232,'a4b'),
        from = c(2,'.*4'),to = 1,
        pattern = c('a:e','b:h'))
df = data.frame(
  a = c(232, 452),
  b = c("nba", "cba")
)
Replace(data = df,
        from = 2,to = 1,
        pattern = c('a:e','b:h'))
```

---

|          |                          |
|----------|--------------------------|
| Replace0 | <i>Replaced by Empty</i> |
|----------|--------------------------|

---

**Description**

Replaced by Empty

**Usage**

Replace0(data, ...)

**Arguments**

data                    can be number, strings, verctors, dataframe or matrix.  
...                    replaced stings

**Value**

replaced data

**Examples**

```
Replace0(data = 232,2)
Replace0(data = c(232,'a4b'),2,'.*4')

df = data.frame(
  a = c(232, 452),
  b = c("nba", "cba")
)
Replace0(data = df, 2,'a')
```

---

|            |                        |
|------------|------------------------|
| Replace_ex | <i>Replace Exactly</i> |
|------------|------------------------|

---

**Description**

Replace Exactly

**Usage**

Replace\_ex(x, from, to, pattern)

**Arguments**

|         |                                            |
|---------|--------------------------------------------|
| x       | vector, dataframe or matrix                |
| from    | replaced stings                            |
| to      | replacements                               |
| pattern | a special pattern, see examples for detail |

**Value**

replaced data

**Examples**

```
a=c(1,2,3,1,4)
Replace_ex(x = a,from = c(1,2),to=5)
Replace_ex(x=a,pattern = c('1:5','2:5'))
Replace_ex(x=a,pattern = '[12]:5')

a=data.frame(v=c(1,2,3,2,4),
             b=c(7,8,9,4,6))
Replace_ex(x = a,from = c(1,2),to=5)
Replace_ex(x=a,pattern = c('1:5','2:5'))
```

---

|           |                                           |
|-----------|-------------------------------------------|
| replicate | <i>Replicate Each Elements of Vectors</i> |
|-----------|-------------------------------------------|

---

**Description**

Replicate Each Elements of Vectors

**Usage**

```
rep_n(x, each)

rep_character(x, each)
```

**Arguments**

|      |                                     |
|------|-------------------------------------|
| x    | vectors                             |
| each | one or more numbers for replication |

**Value**

replicated vectors

Examples

```
rep_n(c('ab', 'cde', 'k', 'op'), 5)
rep_n(c('ab', 'cde', 'k', 'op'), c(4, 6))
rep_n(c('ab', 'cde', 'k', 'op'), c(1, 2, 3, 4))

rep_character(c('ab', 'cde', 'k', 'op'), 5)
rep_character(c('ab', 'cde', 'k', 'op'), c(4, 6))
rep_character(c('ab', 'cde', 'k', 'op'), c(1, 2, 3, 4))
```

---

|                |                                  |
|----------------|----------------------------------|
| reshape_toLong | <i>Convert Wide Data to Long</i> |
|----------------|----------------------------------|

---

Description

It is easy to convert wide data to long in this function. Be careful, id must be unique. prefix, suffix and var.names can be used together.

Usage

```
reshape_toLong(data, prefix = NULL, suffix = NULL, var.names = NULL)
```

Arguments

|           |                                                                                |
|-----------|--------------------------------------------------------------------------------|
| data      | wide data                                                                      |
| prefix    | prefix of value variables                                                      |
| suffix    | suffix of value variables                                                      |
| var.names | names of value variables, do.value will be created as the name of value column |

Value

long data

---

|                |                               |
|----------------|-------------------------------|
| reshape_toWide | <i>Reshape to Wide Format</i> |
|----------------|-------------------------------|

---

Description

Reshape to Wide Format

Usage

```
reshape_toWide(  
  data,  
  key = NULL,  
  value = NULL,  
  prefix = NULL,  
  suffix = NULL,  
  sep = "_"  
)
```

Arguments

|        |                                                     |
|--------|-----------------------------------------------------|
| data   | long data                                           |
| key    | column names for key, which can be one or more      |
| value  | column names for exchange, which can be one or more |
| prefix | column names for prefix, which can be one or more   |
| suffix | column names for suffix, which can be one or more   |
| sep    | seperation                                          |

Value

A wide data.

---

|         |                             |
|---------|-----------------------------|
| reverse | <i>Reverse String Order</i> |
|---------|-----------------------------|

---

Description

Reverse String Order

Usage

```
reverse(x)
```

Arguments

|   |                                  |
|---|----------------------------------|
| x | can be number, strings, verctors |
|---|----------------------------------|

Value

reversed string

Examples

```
reverse(123)  
reverse(c(123, 'abc'))
```

---

right

*Truncate Characters from the Right*


---

**Description**

Truncate Characters from the Right

**Usage**

```
right(x, n)
```

**Arguments**

|   |                                                       |
|---|-------------------------------------------------------|
| x | can be number, strings, vectors, dataframe or matrix. |
| n | length                                                |

**Value**

substring

**Examples**

```
right("abcd",3)
right(c("abc","gjh"),2)
df = data.frame(
  a = c(123,234,456),
  b = c("abc","bcd","hjk")
)
right(df,2)
```

---

right\_equal

*Compare two characters from right Much useful for arguments input.  
Case is ignored.*


---

**Description**

Compare two characters from right Much useful for arguments input. Case is ignored.

**Usage**

```
right_equal(x1, x2)
```

**Arguments**

|    |                     |
|----|---------------------|
| x1 | one character       |
| x2 | the other character |

**Value**

logical

**Examples**

```
right_equal('k', 'OK')
right_equal('ok', 'k')
right_equal('ok', 'Ok')
```

---

|        |                           |
|--------|---------------------------|
| rm_all | <i>Remove all objects</i> |
|--------|---------------------------|

---

**Description**

Remove all objects

**Usage**

```
rm_all()
```

**Value**

empty object

---

|          |                                                |
|----------|------------------------------------------------|
| rm_nchar | <i>Remove elements by number of characters</i> |
|----------|------------------------------------------------|

---

**Description**

Remove elements by number of characters

**Usage**

```
rm_nchar(x, least, most)
```

**Arguments**

- |       |                            |
|-------|----------------------------|
| x     | one vector                 |
| least | least number of characters |
| most  | most number of characters  |

**Value**

removed vector

Examples

```
x <- c('a','abc','abcd',NA)
rm_nchar(x,least = 1)
rm_nchar(x,most = 4)
rm_nchar(x,least = 1, most = 4)
```

---

|          |               |
|----------|---------------|
| row.freq | Row Frequency |
|----------|---------------|

---

Description

Row Frequency

Usage

```
row.freq(x)
```

Arguments

x                      dataframe or matrix

Value

data with frequency column

Examples

```
row.freq(x=mtcars[,8:11])
```

---

|        |                                  |
|--------|----------------------------------|
| select | Subset data Take subset data for |
|--------|----------------------------------|

---

Description

Subset data Take subset data for

Usage

```
select(data, i, ...)

## S3 method for class 'character'
select(data, i, ...)

## S3 method for class 'numeric'
select(data, i, ...)
```

```
## S3 method for class 'logical'
select(data, i, ...)

## S3 method for class 'data.frame'
select(data, i, j, drop = FALSE, ...)

## S3 method for class 'matrix'
select(data, i, j, drop = FALSE, ...)

## S3 method for class 'list'
select(data, i, j, drop = FALSE, ...)
```

### Arguments

|      |                                                                         |
|------|-------------------------------------------------------------------------|
| data | one vector, list, dataframe or matrix                                   |
| i    | element position for vector or list, row number for dataframe or matrix |
| ...  | ignore                                                                  |
| j    | column number for dataframe or matrix                                   |
| drop | logical, whether to drop original format, default is FALSE              |

### Value

selected data

### Examples

```
x <- c('ab', 'bc', 'd')
x |> select(!grepl('a'))
x |> select(grepl('a'))
x |> select(!grepl('a'))
x |> select(grepl('a'))

x <- mtcars
x |> select(!grepl('m', ignore.case = TRUE))
x |> select(grepl('m', ignore.case = TRUE), grepl('m', ignore.case = TRUE))
x |> select(!grepl('m', ignore.case = TRUE), !grepl('m', ignore.case = TRUE))

x |> select(grepl('a') & grepl('m'))
x |> select(grepl('a|m'))
x |> select(am == 1)
```

---

|           |                                     |
|-----------|-------------------------------------|
| seq_range | <i>sequence range of one vector</i> |
|-----------|-------------------------------------|

---

### Description

sequence range of one vector

**Usage**

```
seq_range(x, by = 1)
```

**Arguments**

|    |              |
|----|--------------|
| x  | one vector   |
| by | default is 1 |

**Value**

number sequence

**Examples**

```
seq_range(letters)
seq_range(letters,2)
```

---

|                 |                                |
|-----------------|--------------------------------|
| set_Bioc_mirror | <i>set bioconductor mirror</i> |
|-----------------|--------------------------------|

---

**Description**

set bioconductor mirror

**Usage**

```
set_Bioc_mirror(url)
```

**Arguments**

|     |            |
|-----|------------|
| url | mirror url |
|-----|------------|

**Value**

set bioconductor mirror

---

|                 |                        |
|-----------------|------------------------|
| set_CRAN_mirror | <i>set CRAN mirror</i> |
|-----------------|------------------------|

---

**Description**

set CRAN mirror

**Usage**

```
set_CRAN_mirror(url)
```

**Arguments**

|     |            |
|-----|------------|
| url | mirror url |
|-----|------------|

**Value**

set CRAN mirror

---

|               |                                                                                                  |
|---------------|--------------------------------------------------------------------------------------------------|
| show_function | <i>Show function command line in new script script will be store in your temporary directory</i> |
|---------------|--------------------------------------------------------------------------------------------------|

---

**Description**

Show function command line in new script script will be store in your temporary directory

**Usage**

```
show_function(f, file = NULL)
```

**Arguments**

|      |              |
|------|--------------|
| f    | one function |
| file | file name    |

**Value**

command line in new script

---

|              |                                    |
|--------------|------------------------------------|
| split_expand | <i>Split One Column and Expand</i> |
|--------------|------------------------------------|

---

**Description**

Split One Column and Expand

**Usage**

split\_expand(data, variable, sep)

**Arguments**

- data                dataframe or matrix
- variable           one column name with connected values
- sep                seperated symbol, which can be one or more

**Value**

expanded dataframe or matrix

**Examples**

```
df=data.frame(a=c(1,0),
              b=c('a','n'),
              cyl=c('6;6;4;4;4',
                   '6;8;'))
split_expand(data=df,variable='cyl',sep=';')
```

---

|          |                 |
|----------|-----------------|
| table_NA | <i>Count NA</i> |
|----------|-----------------|

---

**Description**

Count NA

**Usage**

table\_NA(x)

**Arguments**

- x                    object

**Value**

NA and Not count

Examples

```
a <- c(1,2,3,1,NA,NA)
table_NA(a)
```

---

|          |                            |
|----------|----------------------------|
| take_out | <i>Extract Some String</i> |
|----------|----------------------------|

---

Description

Extract Some String

Usage

```
take_out(x, ..., type = "c")
```

Arguments

- x                      string
- ...                    patterns of c('begin','after')
- type                   any left characters of character or list

Value

characters

Examples

```
x='abdghtyu'
take_out(x,c('a','d'),c('h','u'))
```

---

|      |             |
|------|-------------|
| Trim | <i>Trim</i> |
|------|-------------|

---

Description

Trim

Usage

```
Trim(x, pattern = " ")

Trim_left(x, pattern = " ")

Trim_right(x, pattern = " ")
```

**Arguments**

x                      can be vector or dataframe or matrix  
pattern                one or more pattern pattern

**Value**

a trimmed string

---

|              |                          |
|--------------|--------------------------|
| unique_no.NA | <i>Unique Without NA</i> |
|--------------|--------------------------|

---

**Description**

Unique Without NA

**Usage**

unique\_no.NA(x)

**Arguments**

x                      vector

**Value**

unique values with no NA

**Examples**

```
x=c(1,2,3,1,NA)
unique(x)
unique_no.NA(x)
```

---

|          |                       |
|----------|-----------------------|
| unlibray | <i>Detach package</i> |
|----------|-----------------------|

---

**Description**

Detach package

**Usage**

unlibray(x)

**Arguments**

x                      one package name, if missing, detach all packages

**Value**

detach one package

---

|           |                           |
|-----------|---------------------------|
| upper.dir | <i>up level directory</i> |
|-----------|---------------------------|

---

**Description**

up level directory

**Usage**

```
upper.dir(dir, end.slash = TRUE)
```

**Arguments**

dir                      present directory or file path  
end.slash              logical

**Value**

upper directory

---

|            |                                                                                                     |
|------------|-----------------------------------------------------------------------------------------------------|
| write_xlsx | <i>Write data to Excel file write or append one or more data into one Excel file in each sheet.</i> |
|------------|-----------------------------------------------------------------------------------------------------|

---

**Description**

Write data to Excel file write or append one or more data into one Excel file in each sheet.

**Usage**

```
write_xlsx(
  ...,
  file,
  sheet,
  col.names = TRUE,
  row.names = FALSE,
  overwrite = FALSE,
  append = FALSE
)
```

**Arguments**

|           |                                                  |
|-----------|--------------------------------------------------|
| ...       | one or more data                                 |
| file      | Excel file name                                  |
| sheet     | sheet names                                      |
| col.names | logical, whether to write out column names       |
| row.names | logical, whether to write out row names          |
| overwrite | logical, whether to overwrite an existing file   |
| append    | logical, whether to add data to an existing file |

**Value**

write one or more data into one Excel file

**Examples**

```
mtcars2 = mtcars
# write_xlsx(mtcars,mtcars2,file='mtcars')
```

---

|      |                          |
|------|--------------------------|
| %==% | <i>Locate Accurately</i> |
|------|--------------------------|

---

**Description**

Locate Accurately

**Usage**

```
a %==% b
```

**Arguments**

|   |                      |
|---|----------------------|
| a | vector for matching  |
| b | vector for searching |

**Value**

If length of a is one, a vector will be return. If length of a is more than one, a list for each element will be return.

**Examples**

```
a=c(1,2,3,4)
b=c(1,2,3,1,4,1,5,6,1,4,1)
a %==% b
```

---

%+%*Concatenate vectors after converting to character.*

---

**Description**

Concatenate vectors after converting to character.

**Usage**

```
a %+% b
```

**Arguments**

a                    one R objects, to be converted to character vectors.  
b                    one R objects, to be converted to character vectors.

**Value**

one vector

**Examples**

```
1 %+% 1
```

---

%s=%

*Locate Similarly by grep()*

---

**Description**

Locate Similarly by grep()

**Usage**

```
a %s=% b
```

**Arguments**

a                    vector for matching  
b                    vector for searching

**Value**

A list contains location information.

**Examples**

```
1 %s=% c(1,12,3)
c(1,2) %s=% c(1,12,3)
```

# Index

[%+%](#), [63](#)  
[%==%](#), [62](#)  
[%s=%](#), [63](#)

[add\\_biocViews](#), [4](#)  
[all\\_children](#), [4](#)  
[Apriori.Basket](#), [5](#)  
[as.data.frame](#), [6](#)  
[as.transactions](#), [6](#)  
[attr\\_href](#), [7](#)

[c.xml\\_nodeset](#), [7](#)  
[cat\\_n](#), [8](#)  
[character.nms](#), [8](#)  
[chinese\\_utf8](#), [9](#)  
[cnOS](#), [9](#)  
[col\\_split](#), [10](#)  
[columntrans](#), [10](#)  
[compare](#), [11](#)  
[complete.data](#), [12](#)  
[current\\_mirror](#), [12](#)

[decrease](#), [13](#)  
[delete\\_left](#), [13](#)  
[delete\\_up](#), [14](#)  
[deparse0](#), [15](#)  
[desc2df](#), [15](#)  
[dump.it](#), [16](#)  
[dup.connect](#), [16](#)  
[duplicated\\_all](#), [17](#)  
[duplicated\\_last](#), [18](#)

[equal \(compare\)](#), [11](#)  
[equal\\_length](#), [18](#)  
[exec](#), [19](#)  
[expand](#), [20](#)

[factor.it \(columntrans\)](#), [10](#)  
[factor.it<- \(columntrans\)](#), [10](#)  
[factor.nms](#), [20](#)  
[file.dir](#), [21](#)  
[file.name](#), [21](#)  
[fmt](#), [22](#)  
[formal\\_dir](#), [23](#)

[get\\_names](#), [24](#)  
[getBiocmirrors](#), [23](#)  
[give\\_names](#), [24](#)  
[Grepl](#), [25](#)

[has\\_children](#), [26](#)

[in1](#), [27](#)  
[increase](#), [27](#)  
[inner\\_Add\\_Symbol](#), [28](#)  
[insertglue](#), [28](#)  
[install\\_Rversion](#), [29](#)  
[is.dir](#), [29](#)  
[is.linux](#), [30](#)  
[is.mac](#), [30](#)  
[is.windows](#), [31](#)

[join](#), [31](#)  
[join\\_full \(join\)](#), [31](#)  
[join\\_inner \(join\)](#), [31](#)  
[join\\_left \(join\)](#), [31](#)  
[join\\_out \(join\)](#), [31](#)  
[join\\_right \(join\)](#), [31](#)

[keep](#), [32](#)  
[knife](#), [33](#)  
[knife\\_left \(knife\)](#), [33](#)  
[knife\\_right \(knife\)](#), [33](#)

[last](#), [33](#)  
[last\\_column](#), [34](#)  
[last\\_row](#), [34](#)  
[left](#), [35](#)  
[left\\_equal](#), [35](#)  
[legal](#), [36](#)  
[list1](#), [37](#)  
[load\\_extdata](#), [37](#)

lower (compare), 11

mid, 38

mirror.speed, 38

model.data, 39

model.x (model.data), 39

model.y (model.data), 39

NA.col.prob, 40

NA.col.sums, 40

NA.row.prob, 41

NA.row.sums, 41

NA.whole.prob, 42

NA.whole.sums, 42

names\_n, 43

Nchar, 43

numeric.it (columntrans), 10

numeric.it<- (columntrans), 10

numeric.nms, 44

over (compare), 11

package\_all, 44

paste0\_columns, 45

pipe, 45

rd2df, 46

read\_R, 46

rep\_character (replicate), 49

rep\_n (replicate), 49

Replace, 47

Replace0, 48

Replace\_ex, 48

replicate, 49

reshape\_toLong, 50

reshape\_toWide, 50

reverse, 51

right, 52

right\_equal, 52

rm\_all, 53

rm\_nchar, 53

row.freq, 54

select, 54

seq\_range, 55

set\_Bioc\_mirror, 56

set\_CRAN\_mirror, 57

show\_function, 57

split\_expand, 58

table\_NA, 58

take\_out, 59

Trim, 59

trim-left (Trim), 59

trim-right (Trim), 59

Trim\_left (Trim), 59

Trim\_right (Trim), 59

unique\_no.NA, 60

unlibray, 60

upper.dir, 61

write\_xlsx, 61