Package ‘ggloop’

October 20, 2016

Type   Package
Title  Create ‘ggplot2’ Plots in a Loop
Version 0.1.0
URL    https://github.com/seasmith/ggloop
BugReports https://github.com/seasmith/ggloop/issues
Description Pass a data frame and mapping aesthetics to ggloop() in order to create a list of ‘ggplot2’ plots. The way x-y and dots are paired together is controlled by the remapping arguments. Geoms, themes, facets, and other features can be added with the special %L+% (L-plus) operator.
License GPL-2
LazyData TRUE
Imports plyr, ggplot2, magrittr, lazyeval, assertthat
RoxygenNote 5.0.1
Suggests testthat, knitr, rmarkdown
VignetteBuilder knitr
Collate 'utilities.R' 'aes.remap.R' 'utilities.eval2.R'
  'utilities.eval.R' 'aes.wrangle.R' 'aes.loop.R' 'aes.map.R'
  'aes.rename.R' 'ggloop.R' 'lplus.main.R' 'lplus.utilities.R'
NeedsCompilation no
Author Luke Smith [aut, cre]
Maintainer Luke Smith <luke@protocolvital.info>
Repository CRAN
Date/Publication 2016-10-20 01:58:31

R topics documented:
aes_eval ................................................................. 2
aes_group ..................................................................... 3
aes_loop ...................................................................... 4
aes_eval

Assign inputs to x, y or dots.

Description

aes_eval() figures out which variables have been passed and appropriaately assigns the variables to their respective mapping: either (x, y, or dots). Furthermore, it distinguishes between ggplot-like syntax and dplyr-like syntax calling of variables.

Usage

aes_eval(vars, x, y, dots)

Arguments

vars, x, y, dots

Arguments passed from aes_loop() or aes_loop2().

Details

aes_eval() is the first major function to be called by aes_loop().
**aes_group**

**Value**

The list returned by `aes_eval()` is the input for the remapping functions.

The logical vector `$is.dots` is placed between the x and y vectors (if any) and the dots vectors (if any). This is used for easy reference in `if` statements.

The length of each vector (x, y, and dots) in the output list is determined by the length of the vector passed to `aes_loop()`. If an x or y variable is passed more than once, then it will be present in the vector the same number of times it was passed into `aes_loop()`.

**See Also**

Source for names_list and code structure of lazyeval:: function calls can be found at ~/dplyr/R/select-vars.R and ~/dplyr/R/select-utils.R.

---

**aes_group**

Create unique pairings between x, y and dots.

**Description**

`aes_group()` uses a list of x’s and y’s to create each unique combination with dots.

**Usage**

`aes_group(lst)`

**Arguments**

- `lst` A list. The list that will be passed to `aes_group()` will be the list produced by `aes_assing()`.

**Details**

`aes_group()` uses an `lapply` loop to give every dots element with a copy of the x and y vectors (if any). This creates a list in which the first set of components correspond to the combination of dots elements, and the second set of components (the nested components) correspond to the x and y vectors.
aes_loop

Create a list of grouped aesthetic mappings.

Description

aes_loop cannot be used affectively outside of ggloop() (or at least with access to the data frame names).

Usage

aes_loop(x, y, ...)

Arguments

x, y, ... A vector of variable names. Vector can consist of a combination of dplyr-like symbols (unquoted names) and numerics/integers referencing the variable position within data.

Details

aes_loop() is solely meant to be called within ggloop(). To create the raw list of grouped mappings, set ggloop()’s gg_obs argument to FALSE.

Value

aes_loop() returns an environment that includes aes.list (the list of grouped aesthetic mappings used inside ggloop()) and a few vectors used by other functions and lapply() loops for control (to eliminate running duplicate code to return a result from a previously ran function).

cur_vars_env

Helper functions to select NSE (non-standard evaluation) variable names.

Description

Helper functions to select NSE (non-standard evaluation) variable names.

Usage

cur_vars_env

Format

An object of class environment of length 0.
See Also

Source for `select_helpers` and the helper functions can be found at `~/dplyr/R/select-vars.R` and `~/dplyr/R/select-utils.R`.

---

### expand.grid2

**Description**

`expand.grid2()` creates a combination data frame from vectors or lists but differs from the original `expand.grid()` in that it has two options for removing two different type of duplicates. `stringsAsFactors` is set to `TRUE`.

**Usage**

```r
expand.grid2(..., rm.dupes = TRUE, rm.dubs = TRUE)
```

**Arguments**

- `...`: Vectors to be expanded.
- `rm.dupes`: Removes duplicated "rows". If `TRUE` (default) then rows that are unordered duplicates of other rows will be removed. i.e. `c("A", "B", "C")` is the same as `c("C", "B", "A")` and any other combination of "A", "B", and "C".
- `rm.dubs`: Removes a row in which all elements are the same. If `TRUE` (default) then a row such as `c("A", "A", "A")` will be removed.

---

### extract

**Description**

`extract()` simply uses a `for` loop to extract the nth element from each vector in a list. However, it can also operate on a data frame. This is equivalent to taking the first element of each vector and making those elements the first vector in a new list, and it continues on so until it reaches the last element.

**Usage**

```r
extract(lst, num = min(lengths(lst)))
```
Arguments

lst
A list of vectors of equal length, a data frame, or a matrix. If the length of the smallest vector in lst is smaller than num then an error will be thrown (subscript out of bounds).

num
A number (preferably the length of the vectors) to create a sequence for extract() to extract the elements of lst. Default value is the length of the shortest vector in the list.

fun.par
Regular expression pattern for determining if possible function parenthesis are present. Searches for "(" and ")" preceeded by any number of characters.

Description

Regular expression pattern for determining if possible function parenthesis are present. Searches for "(" and ")" preceeded by any number of characters.

Usage

fun.par

Format

An object of class character of length 1.

ggloop
Create ggplot plots in a loop.

Description

ggloop() mimics ggplot() by accepting both a data frame and mappings, returning a plot - or plots in this case. The main difference is that ggloop() accepts vectors for aesthetics and returns a list or nested list of ggplot plots.

Usage

ggloop(data, mappings = aes_loop(), remap_xy = TRUE, remap_dots = FALSE, gg_obs = TRUE, ..., environment = parent.frame())
Arguments

data
Default dataset to use for plot. Must be a data frame and can be only one data frame.
mappings
List of aesthetic mappings to use for plots. Works like mapping from ggplot().
remap_xy
Remapping behavior of the x and y vectors specified in aes_loop(). See details below for more on remapping behavior.
remap_dots
Remapping behavior of the ... vectors specified in aes_loop(). See details below for more on remapping behavior.
gg_obs
Logical. Specifies whether to return plots or the list (or nested list) of aesthetics used to make such a plots.
...
Other arguments. Similar to ggplot()’s ....
environment
An environment and only one environment (cannot be a vector). Similar to ggplot()'s environment.

Details

ggloop() makes use of aes_loop, which is meant to mimic aes from ggplot2. Because of this, the remapping arguments are supplied to ggloop instead of aes_loop().

The first remapping argument, remap_xy can take three values:

- TRUE = The default behavior. All unique combinations of x and y are generated. This means that if a variable (i.e. mpg) is supplied in both x and y, then no mapping will have x and y variables that are the same (i.e. x -> mpg; y -> mpg will not ever happen). Likewise, no unordered pair duplicates will happen (i.e. x -> mpg; y -> cyl and x -> cyl; y -> mpg will be treated the same).
- FALSE = If x and y vectors are not the same length, then the shorter of the two will be recycled. Recycling is similar to mapply()'s recycling.
- NA = If x and y vectors are not the same length, then the shorter of the two will have NA assigned to the missing elements. These are meant to act as placeholders during the wrangling operations (extracting and grouping the aesthetics), and will be taken out before the final list of mappings is sent to ggloop().

Examples

# 1. Return all possible x-y combinations.
plots <- ggloop(data = mtcars,
               mappings = aes_loop(x = mpg:carb, y = mpg:carb))
names(plots)
  # [1] "x.mpg_y.cyl"  "x.mpg_y.disp"  "x.mpg_y.hp"  "x.mpg_y.drat"
  # [5] "x.mpg_y.wt"    "x.mpg_y.qsec"  "x.mpg_y.vs"   "x.mpg_y.am"
  # [9] "x.mpg_y.gear"  "x.mpg_y.carb"  "x.cyl_y.disp" "x.cyl_y.hp"
  # [13] "x.cyl_y.drat"  "x.cyl_y.wt"   "x.cyl_y.qsec" "x.cyl_y.vs"
  # [17] "x.cyl_y.am"    "x.cyl_y.gear"  "x.cyl_y.carb" "x.disp_y.hp"
  # [21] "x.disp_y.drat" "x.disp_y.wt"   "x.disp_y.qsec" "x.disp_y.vs"
  # [25] "x.disp_y.am"   "x.disp_y.gear"  "x.disp_y.carb" "x.hp_y.drat"
  # [29] "x.hp_y.wt"     "x.hp_y.qsec"   "x.hp_y.vs"   "x.hp_y.am"
  # [33] "x.hp_y.gear"   "x.hp_y.carb"  "x.drat_y.wt"  "x.drat_y.qsec"
Plots$x.mpg_y.hp + ggplot2::geom_point()

# 2. Add an additional aesthetic (facet) to plots.
plots2 <- ggloop(data = mtcars,  
                  mappings = aes_loop(  
                                    x = c(disp, hp, wt),  
                                    y = mpg,  
                                    color = factor(cyl))))

# 3. remap_xy = NA  
# The longer vector will go "unpaired" after the shorter vector  
# runs out of elements.
plots3 <- ggloop(data = mtcars,  
                 mappings = aes_loop(x = c(mpg/disp, mpg/hp, mpg/cyl, mpg/gear),  
                                       y = c(hp, disp)),  
                 remap_xy = NA)

# 4. remap_xy = FALSE  
# The longer vector will be "paired" with the shorter vector using  
# recycling (similar to R's internal recycling, i.e. mapply()).
plots4 <- ggloop(data = mtcars,  
                 mappings = aes_loop(x = c(mpg/disp, mpg/hp, mpg/cyl, mpg/gear),  
                                       y = c(hp, disp)),  
                 remap_xy = FALSE)

is.c

Determine if the first element of a parse tree is identical to the c function.

Description

This provides a quick way to evaluate whether the x or y vectors have a c() wrapping. This is important for subsequent subsetting of the respective vectors. Those vectors without a c() wrapping
is.fun

will be wrapped by list(). Symbols are not passed to is.c() due to the subsetting of the first element of the parse-tree.

Usage

is.c(expr)

Arguments

expr A parse tree generated by substitute().

is.fun Is it a function?

Description

Attempts to decipher if a function other than c() has been supplied as input. Returns the position of the possible non-c functions in lst.

Usage

is.fun(lst)

Arguments

lst A list of inputs wrapped in substitute() and coerced to a list using as.list().

is.op Determine if an input uses an arithmetical operator (/, *, -, *, ^).

Description

Matches the argument the ops string using grep. Any matches are subsequently noted and the unique list is returned.

Usage

is.op(lst)

Arguments

lst A list object to be tested.
isFALSE

This is an abbreviation of identical(FALSE, x) to go along with isTRUE()

Description

Use this when needing to test explicitly if a value is FALSE.

Usage

isFALSE(x)

Arguments

x  An object to be tested.

list.pos

Finds the position of a named list element within a list (with no recursion).

Description

All elements in the input list must have a name for this function to give accurate positions. This function can accept a character vector and return the position of each name in the vector.

Usage

list.pos(name, lst)

Arguments

name  A character vector. Ideally a character vector of length 1 (just one name); however it can accept a character vector of length greater than 1. The names in the character vector will be used as names (element headings) in the results vector.

lst  A list with all elements named. If each element does not have a name then there can be no guarantee to the accuracy of the results.

Details

Will return a character vector with names for each element corresponding to the names in the character vector given to the function. If a name is not present in the list then NA is returned.
map_aes

Loop through a list of grouped variables and assign class "uneval" to each element in the group.

Description
This is essentially aes() from ggplot2 placed inside of an lapply() loop. The function name is passed in an mapply() loop inside of aes_loop() and aes_loop2().

Usage
map_aes(lst)

Arguments
lst A list of grouped variables to be assigned class uneval

messy_eval
Reduce the amount of code by turning this sequence into a function.

Description
Reduce the amount of code by turning this sequence into a function.

Usage
messy_eval(expr, vars, names_list)

Arguments
expr Lazy dots.
vars Variable names
names_list List of names built from vars.

Details
The bulk of this code was taken from the dply package.
name_groups  

Extract names for the first level of list components for the returned value of ggloop().

### Description

Extract names for the first level of list components for the returned value of ggloop().

### Usage

```r
dataset <- list(aes = c("a", "b", "c"), xy = c("x", "y"))
name_groups(lst = dataset, dots = c("a", "b", "c", "x", "y"))
```

### Arguments

- `lst`  
  A list - specifically aes.raw.

- `dots`  
  A vector corresponding to the position of the ... arguments in the aes.raw list.

name_subgroups  

Extract names for the second level of list components for the returned value of ggloop().

### Description

Extract names for the second level of list components for the returned value of ggloop().

### Usage

```r
dataset <- list(aes = c("a", "b", "c"), xy = c("x", "y"))
name_subgroups(lst = dataset, dots = c("a", "b", "c", "x", "y"))
```

### Arguments

- `lst`  
  A list - specifically xy.

- `dots`  
  A vector corresponding to the position of the ... arguments in the aes.raw list.
Recycle NA

Description
Will recycle using NA rather than imitating R’s internal recycling mechanism.

Usage
recycle.NA(x, y)

Arguments
x, y Vectors, of which the shorter will be recycled.

Recycle Vector

Description
The shorter of the two vectors will be recycled. Imitates R’s internal recycling mechanism.

Usage
recycle.vector(x, y)

Arguments
x, y Vectors, of which the shorter will be recycled.

Remap XY FALSE

Description
The smallest of the two vectors (x or y) will be recycled in a manner similar to R’s internal recycling mechanism.

Usage
remap_xy_FALSE(1st)
Arguments

lst A list. The list passe will be the raw list generated from calling aes_assign() and is run before a remap function for the any "dots" in the list.

Description

The smallest of the two vectors (x or y) will be recycled with NA instead of using the vector itself (similar to R’s internal recycling mechanism).

Usage

remap_xy_NA(lst)

Arguments

lst A list. The list passe will be the raw list generated from calling aes_assign() and is run before a remap function for the any "dots" in the list.

Description

Matching duplicates (xy pairings that contain identical xy values) will be tossed, and unordered duplicate pairs (xy pairings which match another xy pair (i.e. (mpg, cyl) == (cyl, mpg))) will be tossed.

Usage

remap_xy_TRUE(lst)

Arguments

lst A list. The list passe will be the raw list generated from calling aes_assign() and is run before a remap function for the any "dots" in the list.

Description

Uses expand.grid() to create all possible combinations of xy pairings.
**rm.gg2**

*Remove ggplot2 style and stand-alone aesthetic arguments (i.e. y, x:z, etc).*

**Description**

Expression aesthetics (variables wrapped in functions or using prefix/infix operators) need to be handled differently than just standalone variable aesthetics (i.e. mpg) or *dplyr*-like variable calls (i.e. mpg:hp).

**Usage**

`rm.gg2(expr)`

**Arguments**

- `expr`: A parse tree generated by `substitute()`. If the tree is not wrapped by `c()` then it is advised to wrap `x` with `list()`.

**Details**

The reason it is advised wrap `x` in a `list` is due to the way `x` will be indexed/subsetted. The `c` function wrapping is assumed, so therefore the `list` wrapping is needed.

---

**what**

*Console function for determining: class, type, mode, and names of an object.*

**Description**

Console function for determining: class, type, mode, and names of an object.

**Usage**

`what(x, SIMPLIFY = TRUE)`

**Arguments**

- `x`: An object.
- `SIMPLIFY`: Option to simplify result to a vector (default is TRUE). Result is a list if FALSE.
Add components to a ggloop object.

Description

The `%L+%` (L-plus) operator allows you to add components to a ggloop object - whether that object is a:

- nested list of ggplot plots
- list of ggplot plots
- single ggplot.

Usage

```r
lhs %L+% rhs
```

Arguments

- `lhs` Typically the returned object by `ggloop()`: either a nested list of ggplot objects or a list of ggplot object, but can also be a single ggplot object.
- `rhs` A geom, stat, or other layer feature from the ggplot2 package.

Details

`%L+%` is a substitute for `+` and is used in the same fashion: to add geoms, stats, aesthetics, facets, and other features to `ggplot` object. The returned object from `ggloop()` is often a nested list of `ggplot` objects. However it is possible to use `%L+%` in place of where `+` would normally be used. This is due to the conditional statements present in `%L+%`'s structure.

Examples

```r
# Add component to entire list.
g <- ggloop(mtcars, aes_loop(x = mpg:hp, y = mpg:hp))
g <- g %L+% ggplot2::geom_point()

# Add component to a subset of a list
g2 <- ggloop(mtcars, aes_loop(x = disp:wt, y = disp:wt, color = c(cyl, gear)))
g2$color.gear <- g2$color.gear %L+% ggplot2::geom_point()
g2$color.cyl[1:3] <- g2$color.cyl[1:3] %L+% ggplot2::geom_point()
g2$color.cyl$x.hp_y.drat <- g2$color.cyl$x.hp_y.drat %L+% ggplot2::geom_point()
```
The modified combination of the modulus function (%%) and integer divisor function (%/%).

Description
The placement of the arguments (lhs and rhs) does not matter unlike the actual modulus function (%%) and integer divisor function (%/%).

Usage
lhs %% rhs

Arguments
lhs A number (integer or numeric)
rhs A number (integer or numeric)

The replacement operator. Replaces the lhs with rhs on the condition that length(lhs) == FALSE (the length is 0).

Description
The replacement operator. Replaces the lhs with rhs on the condition that length(lhs) == FALSE (the length is 0).

Usage
lhs %R% rhs

Arguments
lhs An object of any length.
rhs A replacement value if length(lhs) == FALSE.
Index

*Topic datasets
  cur_vars_env, 4
  fun.par, 6
  %L+, 16
  %M%, 17
  %R%, 17
  aes_eval, 2
  aes_group, 3
  aes_loop, 4
  cur_vars_env, 4
  expand.grid2, 5
  extract, 5
  fun.par, 6
  ggloop, 6
  is.c, 8
  is.fun, 9
  is.op, 9
  isFALSE, 10
  list.pos, 10
  map_aes, 11
  messy_eval, 11
  name_groups, 12
  name_subgroups, 12
  recycle.NA, 13
  recycle.vector, 13
  remap_xy_FALSE, 13
  remap_xy_NA, 14
  remap_xy_TRUE, 14
  rm.gg2, 15
  what, 15