Package 'plotscaper'

October 18, 2024

Title Explore Your Data with Interactive Figures

Version 0.2.3

Description A framework for creating interactive figures for data exploration.
All plots are automatically linked and support several kinds of interactive features,
including selection, zooming, panning, and parameter manipulation. The figures
can be interacted with either manually, using a mouse and a keyboard,
or by running code from inside an active R session.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.1)

Imports htmlwidgets (>= 1.6), httpuv (>= 1.6), jsonlite (>= 1.8),
knitr, stats, uuid

Suggests rmarkdown, ggplot2, patchwork, palmerpenguins, dplyr

Config/testthat/edition 3

URL https://bartonicek.github.io/plotscaper/

VignetteBuilder knitr

NeedsCompilation no

Author Adam Bartonicek [aut, cre, cph]

Maintainer Adam Bartonicek <bartonicek@gmail.com>

Repository CRAN

Date/Publication 2024-10-18 10:40:03 UTC

Contents

`add_barplot` *Add a barplot to a scene or schema*

Description

This function adds a barplot to a `plotscaper` scene or schema.

Usage

```
add_barplot(x, variables = NULL, options = NULL)
```

Arguments

`x`	A `plotscaper` scene or schema object
`variables`	A vector of variable names: one discrete (required), one continuous (optional)
`options`	A list of options

Value

The scene or schema back, with the plot added appropriately

See Also

add_plot()

`add_bibarplot` *Add a mirrored barplot to a scene or schema*

Description

This function adds a mirrored barplot to a `plotscaper` scene or schema.

Usage

```
add_bibarplot(x, variables = NULL, options = NULL)
```

Arguments

`x`	A `plotscaper` scene or schema object
`variables`	A vector of variable names: one discrete (required), one or two continuous (required)
`options`	A list of options

Value

The scene or schema back, with the plot added appropriately

See Also

add_plot()

`add_fluctplot` *Add a fluctuation diagram to a scene or schema*

Description

This function adds a fluctuation diagram to a `plotscaper` scene or schema.

Usage

```
add_fluctplot(x, variables = NULL, options = NULL)
```

Arguments

x	A plotscaper scene or schema object
variables	A vector of variable names: two discrete (required), one continuous (optional)
options	A list of options

Value

The scene or schema back, with the plot added appropriately

See Also

add_plot()

---

add_histogram *Add a histogram to a scene or schema*

Description

This function adds a histogram to a plotscaper scene or schema.

Usage

```
add_histogram(x, variables = NULL, options = NULL)
```

Arguments

x	A plotscaper scene or schema object
variables	A vector of variable names: one continuous (required), one continuous (optional)
options	A list of options

Value

The scene or schema back, with the plot added appropriately

See Also

add_plot()

add_histogram2d *Add a 2D histogram to a scene or schema*

Description

This function adds a 2D histogram to a `plotscaper` scene or schema.

Usage

```
add_histogram2d(x, variables = NULL, options = NULL)
```

Arguments

`x`	A `plotscaper` scene or schema object
`variables`	A vector of variable names: two continuous (required), one continuous (optional)
`options`	A list of options

Value

The scene or schema back, with the plot added appropriately

See Also

`add_plot()`

add_pcoords *Add a parallel coordinates plot to a scene or schema*

Description

This function adds a parallel coordinates plot to a `plotscaper` scene or schema.

Usage

```
add_pcoords(x, variables = NULL, options = NULL)
```

Arguments

`x`	A `plotscaper` scene or schema object
`variables`	A vector of variable names: at least two continuous or discrete variables
`options`	A list of options

Value

The scene or schema back, with the plot added appropriately

See Also

add_plot()

---

add_plot *Add a plot to a scene or schema*

Description

This function adds a plot to an existing plotscaper scene or schema. Not meant to be called directly but instead with a wrapper function such as add_scatterplot().

Usage

```
add_plot(x, spec)
```

Arguments

x	A plotscaper scene or schema
spec	A list with the plot specification

Value

The scene or schema back

---

add_scatterplot *Add a scatterplot to a scene or schema*

Description

This function adds a scatterplot to a plotscaper scene or schema.

Usage

```
add_scatterplot(x, variables = NULL, options = NULL)
```

Arguments

x	A plotscaper scene or schema object
variables	A vector of variable names: two continuous or discrete (required), one continuous (optional)
options	A list of options

Value

The scene or schema back, with the plot added appropriately

See Also

add_plot()

---

assigned_cases *Get assigned cases*

Description

This function returns the cases of the data which are assigned to a specific permanent group within a plotscaper scene.

Usage

```
assigned_cases(x, group = 1)
```

Arguments

x	A plotscaper scene
group	The group to retrieve the cases of (can be: 1, 2, or 3)

Value

A numeric vector of case ids

---

assign_cases *Assign cases to a group*

Description

This function assigns specific cases (rows of the data) to a permanent group within a plotscaper scene or schema. Permanent group assignments are only removed by double-clicking.

Usage

```
assign_cases(x, cases = NULL, group = 1)
```

Arguments

x	A plotscaper scene or schema
cases	The cases (rows) to select
group	The group to assign the cases to (can be 1, 2, or 3)

Value

The scene or schema back

clear_layout *Set interactive scene layout*

Description

This function clears an existing layout. See `set_layout()`.

Usage

```
clear_layout(x)
```

Arguments

x	A plotscaper scene

Value

The scene or schema back

create_schema *Create a* plotscaper *schema*

Description

This function constructs a schema of an interactive plotscaper figure.

Usage

```
create_schema(data = NULL, options = NULL)
```

Arguments

data	A dataframe
options	A list of options

Value

An object of class plotscaper_schema

Examples

```
create_schema(mtcars) |> add_scatterplot(c("wt", "mpg")) |> render()
```

dispatch_message *Dispatches a message to a plotscaper scene or schema*

Description

Dispatches a message to a plotscaper scene or schema

Usage

```
dispatch_message(x, message)
```

Arguments

x	A plotscaper scene or schema
message	A list that will get converted to JSON message at appropriate time

Value

The scene or schema back

get_plot_ids *Return a list of plot ids from a plotscaper scene or schema*

Description

Return a list of plot ids from a plotscaper scene or schema

Usage

```
get_plot_ids(x)
```

Arguments

x	A plotscaper scene or schema

Value

The scene or schema back

`get_scale` *Get a plot scale*

Description

This function returns a specific scale from a specifc plot in a `plotscaper` scene.

Usage

```
get_scale(x, id = NULL, scale = NULL)
```

Arguments

`x`	A `plotscaper` scene
`id`	A string id of the plot. See id
`scale`	A string id of the scale (`x`, `y`, `width`, `height`, `area`, or `size`)

Details

This function is primarily meant for internal use, however, you can use it to learn how `plotscaper` implements scales. The output can look a bit overwhelming, however, it's not too complicated once you understand how `plotscaper` scales work.

Each scale has two important properties:

- Domain: The space values are translated *from*
- Codomain: The space values are translated *to*

For example, in a typical scatterplot, the `x` scale might have the range of the data (e.g. `[1, 10]`) as its domain and the width of the plotting region as its codomain (e.g. `[0, 800]` pixels).

The scale's job is to link the domain and codomain, such that we can *push* values forward through the scale, first through the domain and then the codomain. This is done by translating to an intermediate range `[0, 1]`. For example, using the `x` scale above, we might first translate the value `5.5` to `0.5` (midpoint of the domain) and then translate `0.5` to `400` (midpoint of the codomain). We may also be able to reverse the process and *pull* values back through the scale, first through the codomain and then through the domain.

Scale, domain, and codomain each have `props` and `defaults` properties which store the relevant values. For example, for a continuous scale, `props` and `defaults` store the `min` and `max` as well as a transformation function and its inverse (`trans`, `inv`), for a discrete point scale, they store the vector of labels, their order, etc...

On `scale`, the `props` and `defaults` store the following properties: `zero`, `one`, `scale`, `mult`. The `zero` and `one` properties modify where the normalized domain values get placed in the codomain, and vice versa. Suppose our `x` (`[1, 10]`, `[0, 800]` px) scale had `zero = 0.1` and `one = 0.9`. Then data values get pushed to the following intermediate values:

- The value `1` to `0.1` since `0.1 + (1 - 1) / (10 - 1) * (0.9 - 0.1) = 0.1`
- The value `2` to `0.1889` since `0.1 + (2 - 1) / (10 - 1) * (0.9 - 0.1) = 0.1889`

- The value 3 to 0.2778 since 0.1 + ((3 - 1) / (10 - 1)) * (0.9 - 0.1) = 0.2778
- ...
- The value 10 to 0.9 since 0.1 + ((10 - 1) / (10 - 1)) * (0.9 - 0.1) = 0.9

When those values get translated to the space of the codomain, we end up with 10% margins on each side, i.e.

- The value 1 gets pushed to 80 pixels
- ...
- The value 10 gets pushed to 720 pixels

The scale and mult properties both multiply the normalized domain values. They work the same way, however, they are different semantically: scale is meant to be constant whereas mult may change dynamically, through interaction. For example, by default, in a barplot, the width scale gets assigned the scale value of 1 / k, where k is the number of categories/bars, and a mult value of 0.9. This means that each bar is 1 / k * 0.9 * [plot width in pixels] wide, and we can dynamically make it wider or narrower by pressing the +/- keys to modify the mult property (but not the scale property).

Value

A list of scale properties

---

id *Plot id*

Description

A string which uniquely identifies a plot plotscaper scene or schema.

Usage

```
id
```

Format

An object of class NULL of length 0.

Details

id is a string that uniquely identifies a plot within a plotscaper scene or schema. It can match a plot based on its position (e.g. "plot1", "plot2", ...), in the order the plots were added, left-to-right top-to-bottom, or it can match plot based on type (e.g. "scatter1" or "barplot3"), again, in order of addition.

If the plot is matched based on type, the morphemes "plot" and "gram" are ignored, such that e.g. "scatterplot1" is the same as "scatter1" and "histogram2d4" is the same as "histo2d4".

The string can also be shortened, e.g. "p1" for "plot1", "s2" for "scatter2", or "hh3" for "histo2d3".

normalize *Normalize a plot*

Description

This function switches the representation of a plot to a normalized one, e.g. spineplot, spinogram, etc...

Usage

```
normalize(x, id = NULL)
```

Arguments

x	A plotscaper scene
id	A string id of the plot. See id

Value

The scene or schema back

plotscaper-shiny *Shiny bindings for plotscaper*

Description

Output and render functions for using plotscaper within Shiny applications and interactive Rmd documents.

Usage

```
plotscaperOutput(outputId, width = "100%", height = "400px")

renderPlotscaper(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

outputId	output variable to read from
width, height	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
expr	An expression that generates a plotscaper
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

Value

A Shiny widget output

A rendered Shiny widget

`plotscaper_global` *A Plotscaper Global Configuration Object*

Description

Used mainly for setting up the HTTP server for communication between an interactive R session and the figure.

Usage

```
plotscaper_global
```

Format

An object of class `environment` of length 7.

`pop_plot` *Remove the last plot from a scene or schema*

Description

This function removes the last plot from a `plotscaper` scene or schema.

Usage

```
pop_plot(x)
```

Arguments

`x`	A plotscaper scene or schema

Value

The scene or schema back

reducer *Create a reducer*

Description

Constructs a reducer that can be used to show alternative summaries in a `plotscaper` plot.

Usage

```
reducer(initialfn = NULL, reducefn = NULL, name = NULL)
```

Arguments

`initialfn`	An JavaScript initializing function
`reducefn`	A JavaScript reducer function specified
`name`	A name for the reducer (a string)

Details

`reducefn` and `initialfn` should be strings interpretable as JavaScript functions. Further:

- `initialfn` should take 0 arguments and just return some value (i.e. a thunk).
- `reducefn` should take two arguments `previous` and `next` and return a result of the same type as `previous`.

Value

A reducer (which is really just a `list` with some additional formatting)

Examples

```
r <- reducer(initialfn = "() => 0",
             reducefn = "(x, y) => Math.max(x, y)",
             name = "max")
create_schema(mtcars) |>
  add_barplot(c("cyl", "mpg"), options = list(reducer = r))
```

remove_plot *Remove specific plot from a scene or schema*

Description

This function removes a specific plot from a `plotscaper` scene or schema.

Usage

```
remove_plot(x, id = NULL)
```

Arguments

`x`	A plotscaper scene or schema
`id`	A string id of the plot. See id

Value

The scene or schema back

render *Render a schema into an interactive scene*

Description

This function takes a `plotscaper` schema and renders it as a concrete `htmlwidgets` widget.

Usage

```
render(
  schema,
  launch_server = TRUE,
  width = NULL,
  height = NULL,
  elementId = NULL,
  options = NULL
)
```

Arguments

`schema`	A `plotscaper` schema object
`launch_server`	Whether to launch a httpuv server for interaction with figure
`width`	Width
`height`	Height
`elementId`	Id of the HTML element to render the scene in (optional)
`options`	A list of options

Value

An object of class `plotscaper_scene`

---

reset *Reset a scene or schema*

Description

This function resets a `plotscaper` scene or schema. All selection/group assignment will be removed, and axis limits/levels of zoom will be restored to default.

Usage

```
reset(x)
```

Arguments

`x`	A `plotscaper` scene or schema

Value

The scene or schema back

---

selected_cases *Get selected cases*

Description

This function returns the cases of the data which are selected within a `plotscaper` scene.

Usage

```
selected_cases(x)
```

Arguments

`x`	A `plotscaper` scene

Value

A numeric vector of case ids

select_cases *Select cases of the data*

Description

This function selects specific cases (rows of the data) within a `plotscaper` scene or schema by assigning them to transient selection. Transient group assignment is removed by clicking.

Usage

```
select_cases(x, cases = NULL)
```

Arguments

`x`	A `plotscaper` scene or schema
`cases`	The cases (rows) to select

Value

The scene or schema back

set_layout *Set interactive scene layout*

Description

This function sets a layout for a `plotscaper` scene. Similar to the `graphics::layout` function.

Usage

```
set_layout(x, layout = NULL)
```

Arguments

`x`	A `plotscaper` scene
`layout`	A numeric matrix of plot ids, arranged into contiguous rectangles

Value

The scene or schema back

`set_parameters` *Set reactive parameters*

Description

This functions sets reactive paramaters on a plot such as a histogram.

Usage

```
set_parameters(
  x,
  id = NULL,
  width = NULL,
  anchor = NULL,
  width_x = NULL,
  anchor_x = NULL,
  width_y = NULL,
  anchor_y = NULL
)
```

Arguments

`x`	A `plotscaper` scene
`id`	A string id of the plot. See id
`width`	Histogram binwidth
`anchor`	Histogram anchor
`width_x`	2D histogram binwidth (x-axis)
`anchor_x`	2D histogram anchor (x-axis)
`width_y`	2D histogram binwidth (y-axis)
`anchor_y`	2D histogram anchor (y-axis)

Value

The scene or schema back

`set_scale` *Set values of a scale*

Description

This function sets the values of a scale within one plot inside a `plotscaper` scene or schema.

Usage

```
set_scale(
  x,
  id = NULL,
  scale = NULL,
  min = NULL,
  max = NULL,
  breaks = NULL,
  zero = NULL,
  one = NULL,
  direction = NULL,
  mult = NULL,
  default = NULL,
  unfreeze = NULL
)
```

Arguments

`x`	A `plotscaper` scene or schema
`id`	A string id of the plot. See id
`scale`	A string identifying scale. Can be: "x", "y", "area", or "size".
`min`	Scale minimum (continuous scales only)
`max`	Scale maximum (continuous scales only)
`breaks`	A vector of discrete breaks (discrete scale only)
`zero`	The proportion of codomain to which the smallest/first value gets mapped to
`one`	The proportion of codomain to which largest/last value gets mapped to
`direction`	Scale direction. Can be `1` or `-1`
`mult`	Scale multiplier
`default`	Whether to set other arguments as scale defaults
`unfreeze`	Whether to unfreeze frozen parameters (such as the lower y-axis limit in barplot)

Value

The scene or schema back

start_server — *Start a server for communication between the R session and* plostcaper *scenes*

Description

This function starts an httpuv server for an interactive communication between the R session and plotscaper scenes. Uses plotscaper_global options.

Usage

```
start_server(random_port = FALSE)
```

Arguments

random_port	Whether to use a random port number. Useful if the default port is already taken.

Value

Nothing (called for side effects)

zoom — *Zoom into an area of a plot*

Description

This function zooms into a rectangular area of the specified plot. The coordinates of the rectangular area can be specified with either percentages of the plotting region, absolute coordinates (pixels), or data coordinates.

Usage

```
zoom(x, id = NULL, coords = NULL, units = "pct")
```

Arguments

x	A plotscaper scene or schema
id	A string id of the plot. See id
coords	The coordinates of a rectangle to zoom into, in the following order: x0, y0, x1, y1
units	The units with which to interpret the coordinates. Can be "pct" (percentages of the plotting region), "abs" (absolute screen coordinates, in pixels), or "data" (data coordinates; only works if both scales are continuous).

Value

The scene or schema back

Index