Package: trelliscope (via r-universe)

August 20, 2024

Title Create Interactive Multi-Panel Displays

Version 0.1.14

Description Trelliscope enables interactive exploration of data frames of visualizations.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Suggests testthat (>= 3.0.0), covr, plotly, fidelius, dygraphs, rmarkdown

Config/testthat/edition 3

Imports R6, jsonlite, ggplot2 (>= 3.2.1), dplyr, rlang, svglite, htmlwidgets, htmltools, cli, tidyr, httpuv, knitr, digest, glue, magick, shiny, pillar, vctrs, tibble

Depends R (>= 4.1)

LazyData true

URL https://trelliscope.org/trelliscope, https://github.com/trelliscope/trelliscope

BugReports https://github.com/trelliscope/trelliscope/issues

VignetteBuilder knitr

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

RemoteUrl https://github.com/trelliscope/trelliscope

RemoteRef HEAD

RemoteSha 99dfb29c0057680ecf92ed2b061dd833cfd667f8

Contents

`add_charm` *Use fidelius to password protect a trelliscope display*

Description

Use fidelius to password protect a trelliscope display

Usage

```
add_charm(trdf, ...)
```

Arguments

`trdf`	A trelliscope data frame created with `as_trelliscope_df()` or a data frame which will be cast as such.
`...`	Arguments passed to the charm() function in the fidelius package.

`add_inputs` *Add inputs to a trelliscope display*

Description

Add inputs to a trelliscope display

Usage

```
add_inputs(trdf, ..., email = NULL, vars = NULL)
```

Arguments

`trdf`	A trelliscope data frame created with `as_trelliscope_df()` or a data frame which will be cast as such.
`...`	Any number of input specifications. These can be specified with any of `input_number()`, `input_radio()`, `input_checkbox()`, `input_select()`, `input_multiselect()`, `input_text()`
`email`	An email address (optional).
`vars`	A vector of meta variable names found in the display. These will be made available as columns in the csv download of user inputs.

add_trelliscope_resource_path *Add Trelliscope resource path for Shiny app*

Description

Add Trelliscope resource path for Shiny app

Usage

```
add_trelliscope_resource_path(prefix, path)
```

Arguments

prefix, path	See shiny::addResourcePath()

add_view *Add a view specification to a trelliscope display*

Description

Add a view specification to a trelliscope display

Usage

```
add_view(trdf, name, ...)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
name	The name of the view.
...	Any number of state specifications that define the view. These can be specified with any of state_layout(), state_labels(), state_sort(), filter_string(), filter_range().

as_json *Convert any trelliscope R6 object to JSON*

Description

Convert any trelliscope R6 object to JSON

Usage

```
as_json(trdf, pretty = TRUE)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
pretty	Adds indentation whitespace to JSON output. Can be TRUE/FALSE or a number specifying the number of spaces to indent.

as_panels_df *Render the panels of a trelliscope display*

Description

Render the panels of a trelliscope display

Usage

```
as_panels_df(
  x,
  panel_col = "panel",
  keep_cols = NULL,
  as_plotly = FALSE,
  plotly_args = NULL,
  plotly_cfg = NULL
)
```

Arguments

x	A ggplot object created with facet_panels().
panel_col	The name of the column to store the rendered panels in.
keep_cols	An optional vector of extra variable names in x to keep in the data. If specified, its values cannot vary within each combination of the specified facet variables.
as_plotly	Should the panels be written as plotly objects?
plotly_args	Optional named list of arguments to send to ggplotly
plotly_cfg	Optional named list of arguments to send to plotly's 'config" method.

`as_trelliscope_df` *Instantiate a trelliscope data frame*

Description

Instantiate a trelliscope data frame

Usage

```
as_trelliscope_df(
  df,
  name = NULL,
  description = name,
  key_cols = NULL,
  tags = NULL,
  path = NULL,
  force_plot = FALSE,
  key_sig = NULL,
  order = 0,
  jsonp = TRUE
)
```

Arguments

`df`	A data frame that contains the metadata of the display as well as a column that indicate the panels to be displayed.
`name`	Name of the trelliscope display.
`description`	Description of the trelliscope display.
`key_cols`	Variable names in `df` that uniquely define a row of the data. If not supplied, an attempt will be made to infer them.
`tags`	Optional vector of tag names to identify the display in the case that there are many to search through.
`path`	Directory in which to place the trelliscope display when it is written using `write_trelliscope()`.
`force_plot`	Should the panels be forced to be plotted, even if they have already been plotted and have not changed since the previous plotting?
`key_sig`	A string "signature" that represents the panels for this display. This should not be specified unless you know what you are doing.
`order`	If there will be multiple displays in the same directory, this can be used to specify the order in which they should be listed. The display with the lowest order will be shown on load.
`jsonp`	If true, app data files are written as "jsonp" format, otherwise "json" format. The "jsonp" format makes it possible to browse a trelliscope app without the need for a web server and is set to TRUE by default. Use "json" if you are deploying to a web server that doesn't work well with "jsonp".

Examples

```
# Use `as_trelliscope_df()` to convert panel metadata to a special
# trelliscope data frame
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
  ) |>
    as_panels_df()

meta_dat <- gap |>
  group_by(country, continent) |>
  summarise(
    mean_life_exp = mean(life_exp),
    min_life_exp = min(life_exp),
    max_life_exp = max(life_exp),
    mean_gdp = mean(gdp_percap),
    .groups = "drop"
  )

joined_dat <- left_join(panel_dat, meta_dat) |>
  as_trelliscope_df(name = "life_expectancy", path = tempfile())

## Not run:
view_trelliscope(joined_dat)

## End(Not run)

# You can also use `as_trelliscope_df()` on datasets that have links to
# images instead of conventional ggplot objects
## Not run:
```

---

currencies *Data frame of currencies and their codes*

Description

Data frame of currencies and their codes

Usage

```
currencies
```

Format

An object of class `spec_tbl_df` (inherits from `tbl_df`, `tbl`, `data.frame`) with 278 rows and 5 columns.

Source

https://www.six-group.com/en/products-services/financial-information/data-standards.html#scrollTo=maintenance-agency

---

`currency` *Create a currency vector*

Description

Create a currency vector

Usage

```
currency(x = double(), code = "USD", digits = 2, locale = TRUE, log = NULL)
```

Arguments

`x`	A numeric vector.
`code`	Currency code. See currencies for a list of possibilities.
`digits`	How many digits to round to when displaying the number. If `NULL`, all digits will be shown. Default is 2.
`locale`	Should the variable be displayed using its locale?
`log`	Should the variable's distribution be shown on the log scale? If not specified, an inference will be made based on its values.

`facet_panels` *Add a trelliscope facet to a ggplot*

Description

Add a trelliscope facet to a ggplot

Usage

```
facet_panels(
  facets,
  scales = "same",
  add_plot_metrics = FALSE,
  unfacet = c("none", "line", "point"),
  unfacet_col = "gray",
  unfacet_alpha = 0.4,
  data = ggplot2::waiver()
)
```

Arguments

facets	A formula to facet the panels on. Similar to ggplot2::facet_wrap()'s 'facets"
scales	Should scales be the same ("same", the default), free ("free"), or sliced ("sliced"). May provide a single string or two strings, one for the X and Y axis respectively.
add_plot_metrics	Should metrics about each panel be automatically calculated? These metrics are based on the context of what is being plotted, e.g. correlation coefficient if plot is a scatterplot.
unfacet	Specifies whether to "unfacet" the data such that all of the data appears in the background of the plot. Options are "none" (default), "line" or "point". The latter two options will add either a line or point layer, grouped by the faceting variables, underneath each panel. This is useful for time series plots for viewing each panel in relation to others.
unfacet_col	The color to use for the "unfacet" lines or points.
unfacet_alpha	The alpha to use for the "unfacet" lines or points.
data	data used for faceting. Defaults to the main data argument to ggplot2::ggplot().

Examples

```
# You can run facet_panels() just like how you would run facet_wrap()
library(ggplot2)

## Not run:
ggplot(gap, aes(year, life_exp)) +
  geom_point() +
  facet_panels(vars(country, continent))

## End(Not run)

# facet_panels can also be a jumping off point into setting up a more
# developed trelliscope by passing into `as_panels_df()` to create a nested
# trelliscope data frame for additional editing.
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
  geom_point() +
  facet_panels(vars(country, continent))
) |>
  as_panels_df()

trell_df <- panel_dat |>
  as_trelliscope_df(name = "life expectancy", path = "gapminder") |>
  set_default_layout(ncol = 4)

## Not run:
view_trelliscope(trell_df)

## End(Not run)
```

filter_cat_href *Create a link to another display filtered to categories of a variable*

Description

Create a link to another display filtered to categories of a variable

Usage

```
filter_cat_href(
  display_name,
  varname,
  values = NULL,
  labels = NULL,
  prefix = NULL
)
```

Arguments

display_name	The name of the display to link to.
varname	The name of the variable to filter on.
values	The values of the variable to filter on.
labels	Labels in the linked display to show.
prefix	A prefix to add to the link. Not needed if the display is part of the same app.

filter_range *Specify a range "filter" state (applies to numeric, date, and datetime meta variables)*

Description

Specify a range "filter" state (applies to numeric, date, and datetime meta variables)

Usage

```
filter_range(varname, min = NULL, max = NULL)
```

Arguments

varname	The name of the variable to filter on.
min	Lower bound of the range (if NULL, there is no lower bound).
max	Upper bound of the range (if NULL, there is no upper bound).

filter_string — *Specify a string "filter" state (applies to string and factor meta variables)*

Description

Specify a string "filter" state (applies to string and factor meta variables)

Usage

```
filter_string(varname, regexp = NULL, values = NULL)
```

Arguments

varname	The name of the variable to filter on.
regexp	A search string that can be a regular expression indicating the values of the variable to filter on.
values	A vector of specific values of the variable to filter on. If values is specified, regexp will be ignored.

gap — *Gapminder data with ISO codes and country centroids*

Description

Gapminder data with ISO codes and country centroids

Usage

```
gap
```

Format

An object of class tbl_df (inherits from tbl, data.frame) with 1704 rows and 9 columns.

Source

https://cran.r-project.org/web/packages/gapminder/index.html

href *Create a vector of links*

Description

Create a vector of links

Usage

```
href(x = character())
```

Arguments

x	A character vector.

input_checkbox *Specify a "checkbox" input*

Description

Specify a "checkbox" input

Usage

```
input_checkbox(name, label = name, active = TRUE, options)
```

Arguments

name	Name of the input.
label	Description of the input.
active	Should the input be active by default?
options	A vector of checkbox options.

See Also

Other inputtypes: input_multiselect(), input_number(), input_radio(), input_select(), input_text()

Examples

```
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
  ) |>
  as_panels_df()

trell <- panel_dat |>
as_trelliscope_df() |>
add_inputs(
  input_checkbox(
    name = "Checkbox Input",
    label = "A space to add custom button inputs",
    options = c("yes", "no")
  )
)
## Not run:
view_trelliscope(trell)

## End(Not run)
```

---

input_multiselect *Specify a "multiselect" input*

Description

Specify a "multiselect" input

Usage

```
input_multiselect(name, label = name, active = TRUE, options)
```

Arguments

name	Name of the input.
label	Description of the input.
active	Should the input be active by default?
options	A vector of options for the multiselect dropdown.

See Also

Other inputtypes: input_checkbox(), input_number(), input_radio(), input_select(), input_text()

Examples

```
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
  ) |>
  as_panels_df()

trell <- panel_dat |>
as_trelliscope_df() |>
add_inputs(
  input_multiselect(
    name = "Multiselect Input",
    label = "A space to add custom dropdown inputs",
    options =c("yes", "no")
  )
)
## Not run:
view_trelliscope(trell)

## End(Not run)
```

`input_number` *Specify a "numeric" input box*

Description

Specify a "numeric" input box

Usage

```
input_number(name, label = name, active = TRUE, min = NULL, max = NULL)
```

Arguments

`name`	Name of the input.
`label`	Description of the input.
`active`	Should the input be active by default?
`min`	Optional minimum value to allow in the input.
`max`	Optional maximum value to allow in the input.

See Also

Other inputtypes: `input_checkbox()`, `input_multiselect()`, `input_radio()`, `input_select()`, `input_text()`

Examples

```
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
  ) |>
  as_panels_df()

trell <- panel_dat |>
as_trelliscope_df() |>
add_inputs(
  input_number(
    name = "Numeric Input",
    label = "A space to add custom ranking for sorting",
    min = 0, max = 10
  )
)
## Not run:
view_trelliscope(trell)

## End(Not run)
```

`input_radio` *Specify a "radio button" input*

Description

Specify a "radio button" input

Usage

```
input_radio(name, label = name, active = TRUE, options)
```

Arguments

`name`	Name of the input.
`label`	Description of the input.
`active`	Should the input be active by default?
`options`	A vector of radio button options.

See Also

Other inputtypes: `input_checkbox()`, `input_multiselect()`, `input_number()`, `input_select()`, `input_text()`

Examples

```
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
  ) |>
  as_panels_df()

trell <- panel_dat |>
as_trelliscope_df() |>
add_inputs(
  input_radio(
    name = "Radio Input",
    label = "A space to add custom ranking for sorting",
    options = c("yes", "no")
  )
)
## Not run:
view_trelliscope(trell)

## End(Not run)
```

`input_select` *Specify a "select" input*

Description

Specify a "select" input

Usage

```
input_select(name, label = name, active = TRUE, options)
```

Arguments

`name`	Name of the input.
`label`	Description of the input.
`active`	Should the input be active by default?
`options`	A vector of options for the select dropdown.

See Also

Other inputtypes: `input_checkbox()`, `input_multiselect()`, `input_number()`, `input_radio()`, `input_text()`

Examples

```
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
  ) |>
  as_panels_df()

trell <- panel_dat |>
as_trelliscope_df() |>
add_inputs(
  input_select(
    name = "Select Input",
    label = "A space to add custom dropdown inputs",
    options =c("yes", "no")
  )
)
## Not run:
view_trelliscope(trell)

## End(Not run)
```

`input_text` *Specify a "text box" input*

Description

Specify a "text box" input

Usage

```
input_text(name, label = name, active = TRUE, height = 3)
```

Arguments

`name`	Name of the input.
`label`	Description of the input.
`active`	Should the input be active by default?
`height`	Height (in lines of text) of the text box input.

See Also

Other inputtypes: `input_checkbox()`, `input_multiselect()`, `input_number()`, `input_radio()`, `input_select()`

Examples

```
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
  ) |>
  as_panels_df()

trell <- panel_dat |>
  as_trelliscope_df() |>
  add_inputs(
    input_text(
      name = "Text Input",
      label = "A space to add custom text input"
    )
)
## Not run:
view_trelliscope(trell)

## End(Not run)
```

`mars_rover` *Mars rover data*

Description

Mars rover data

Usage

```
mars_rover
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1245 rows and 11 columns.

Source

https://github.com/corincerami/mars-photo-api

number *Create a number vector*

Description

Create a number vector

Usage

```
number(x = double(), digits = 2, locale = TRUE, log = NULL)
```

Arguments

x	A numeric vector.
digits	How many digits to round to when displaying the number. If not specified, a value will be inferred. If -1, all digits will be shown.
locale	Should the variable be displayed using its locale? For example, 1234.56 in US would be displayed as 1,234.56.
log	Should the variable's distribution be shown on the log scale? If not specified, an inference will be made based on its values.

panel_lazy *Add a "calculated panel" column to a dataset*

Description

Add a "calculated panel" column to a dataset

Usage

```
panel_lazy(plot_fn, data = dplyr::pick(dplyr::everything()))
```

Arguments

plot_fn	A function that produces a panel from a given subset of data, taking as arguments any variable names in data that you would like to be available in the plottinf function.
data	A data frame from which subsets will be extracted and plots will be made. Should be a superset of the summary dataset to which this plot column is being added.

panel_local *Add a "panel_local" column to a dataset*

Description

Add a "panel_local" column to a dataset

Usage

```
panel_local(x = character(), type = NULL)
```

Arguments

x	A character vector of paths to local files to be used as panels.
type	The "type" of panel ("img" or "iframe"). If NULL, will be inferred from the file extension.

panel_options *Specify options for lazily-rendered panels in a Trelliscope display*

Description

Specify options for lazily-rendered panels in a Trelliscope display

Usage

```
panel_options(
  width = NULL,
  height = NULL,
  format = NULL,
  force = FALSE,
  prerender = TRUE
)
```

Arguments

width	Width in pixels of each panel.
height	Height in pixels of each panel.
format	The format of the panels the server will provide. Can be one of "png" , "svg", or "html". Ignored if panel is not lazy.
force	Should server force panels to be written? If FALSE, if the panel has already been generated, that is what will be made available. Ignored if panel is not lazy.
prerender	If "TRUE", lazy panels will be rendered prior to viewing the display. If "FALSE", a local R websockets server will be created and plots will be rendered on the fly when requested by the app. The latter is only available when using Trelliscope locally. Ignored if panel is not lazy.

Examples

```
mars_rover |>
  as_trelliscope_df(name = "mars rover") |>
  set_panel_options(img_src = panel_options(width = 2, height = 1))
# TODO
```

`panel_url` *Add a "panel_url" column to a dataset*

Description

Add a "panel_url" column to a dataset

Usage

```
panel_url(urls, type = NULL)
```

Arguments

`urls`	A character vector of URLs to be used as panels.
`type`	The "type" of panel ("img" or "iframe"). If NULL, will be inferred from the file extension.

`rover_icon_b64` *Base-64 encoded icon of a rover for use in the Mars rover example*

Description

Base-64 encoded icon of a rover for use in the Mars rover example

Usage

```
rover_icon_b64
```

Format

An object of class `character` of length 1.

set_default_filters *Add a filter state specifications to a trelliscope display*

Description

Add a filter state specifications to a trelliscope display

Usage

```
set_default_filters(trdf, ..., add = TRUE)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
...	Filter state specifications (e.g. filter_string(), filter_range()).
add	Should existing filter state specifications be added to? Default is TRUE. If FALSE, the entire sort specification will be overridden.

set_default_labels *Add a labels state specification to a trelliscope display*

Description

Add a labels state specification to a trelliscope display

Usage

```
set_default_labels(trdf, varnames)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
varnames	A vector of variable names whose values should appear as labels. If NULL, no labels will be displayed. when viewing the display.

set_default_layout *Add a layout state specification to a trelliscope display*

Description

Add a layout state specification to a trelliscope display

Usage

```
set_default_layout(
  trdf,
  ncol = 1,
  page = 1,
  sidebar = FALSE,
  visible_filters = NULL
)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
ncol	Number of columns of panels to show.
page	The page number to show.
sidebar	Should the sidebar be shown?
visible_filters	A vector of variable names to add as visible filters in the sidebar.

set_default_sort *Add a labels state specification to a trelliscope display*

Description

Add a labels state specification to a trelliscope display

Usage

```
set_default_sort(trdf, varnames, dirs = "asc", add = FALSE)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
varnames	A vector of variable names to sort on.
dirs	A vector of directions to sort on ("asc" or "desc").
add	Should an existing sort specification be added to? If FALSE (default), the entire sort specification will be overridden.

set_info_html *Specify custom "display info" html*

Description

Specify custom "display info" html

Usage

```
set_info_html(trdf, file)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
file	Path to an existing html file to use.

set_panel_options *Set panel options for a Trelliscope display*

Description

Set panel options for a Trelliscope display

Usage

```
set_panel_options(trdf, ...)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df()
...	A named set panel options to set. The names should correspond to the names of the variables in the data frame. The values should come from panel_options().

Examples

```
mars_rover |>
  as_trelliscope_df(name = "mars rover") |>
  set_panel_options(img_src = panel_options(width = 2, height = 1))
```

set_primary_panel *Set the primary panel of a trelliscope display*

Description

Set the primary panel of a trelliscope display

Usage

```
set_primary_panel(trdf, name)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
name	The name of the panel to set as the primary panel that will be viewed when launching the app.

set_show_info_on_load *Show "display info" when display first loads*

Description

Show "display info" when display first loads

Usage

```
set_show_info_on_load(trdf, show = TRUE)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df() or a data frame which will be cast as such.
show	Should display info be shown on load?

set_tags *Set tags for variables in a data frame*

Description

Set tags for variables in a data frame

Usage

```
set_tags(d, ...)
```

Arguments

d	A data frame.
...	A named set of vectors of variable names, where each name is a tag and each value in each vector corresponds to one of the variables in the dataset.

Examples

```
set_tags(iris,
  info = "Species",
  metrics = c("Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width"),
  width = c("Sepal.Width", "Petal.Width"))
```

set_theme *Set a color theme for a Trelliscope display*

Description

Set a color theme for a Trelliscope display

Usage

```
set_theme(
  trdf,
  primary = "#448aff",
  primary2 = "#4dabf5",
  primary3 = "#2e60b1",
  background = "#FEFEFE",
  background2 = "#EBEBEB",
  background3 = "#E0E0E0",
  bars = "#FFAE25",
  text = "#000",
  text2 = "#FFF",
  button_text = "#757575",
  text_disabled = "#BCBCBC",
```

```
  error = "#ff5252",
  font_family = "\"Poppins\", sans-serif",
  logo = NULL
)
```

Arguments

trdf	A trelliscope data frame created with as_trelliscope_df()
primary	The primary color which applies to the main controls in the app (buttons, navigation icons, checkboxes, etc.).
primary2	The second-level primary color which applies to hover states, etc. Typically a slightly lighter shade of the primary color.
primary3	The third-level primary color, mainly used for the fullscreen button. Typically a slightly darker shade of the primary color.
background	The background color of the app.
background2	The second-level background color used in various components and headers. Typically a slightly lighter or darker shade of the background color.
background3	The third-level background color used for inactive filter bars, sub-header background, component outlines, etc. Typically a slightly lighter or darker shade of the background color.
bars	Color used for highlighted bars in the filter graphs (histogram and bar chart).
text	The general text color used in the app.
text2	The secondary text color used in the header, etc. Should be a contrasting color to the primary color.
button_text	The text color used for buttons. Should be a contrasting color to the primary color.
text_disabled	The text color used for disabled buttons, etc.
error	The color used for error messages.
font_family	The font family to use in the app. Default is "Poppins". Note that many aspects of the app are styled with this font, so changing it may result in less-attractive styling.
logo	URL (relative or absolute) to a logo image to include in the header.

Examples

```
x <- mars_rover |>
  as_trelliscope_df(name = "mars rover") |>
  set_theme(
    primary = "#c80000",
    primary2 = "#f00000",
    primary3 = "#960000",
    background = "#222222",
    background2 = "#444444",
    background3 = "#333333",
    bars = "#c80000",
```

```
    text = "#ffffff",
    text2 = "#ffffff",
    text_disabled = "#bcbcbc",
    logo = rover_icon_b64
  )
```

`set_var_labels` *Set labels for variables in a data frame*

Description

Set labels for variables in a data frame

Usage

```
set_var_labels(d, ...)
```

Arguments

`d`	A data frame.
`...`	A named set of labels, where each name must correspond to one of the variables in the dataset.

Examples

```
set_var_labels(mtcars, mpg = "Miles per gallon", cyl = "Number of cylinders")
```

`show_info` *View trelliscope info of a trelliscope data frame*

Description

View trelliscope info of a trelliscope data frame

Usage

```
show_info(trdf)
```

Arguments

`trdf`	A trelliscope data frame created with `as_trelliscope_df()` or a data frame which will be cast as such.

Examples

```
library(ggplot2)
library(dplyr)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(~country + continent)
  ) |>
  as_panels_df()

trell <- panel_dat |>
  as_trelliscope_df(name = "life expectancy", path = "gapminder") |>
  set_default_layout(ncol = 4)

show_info(trell)
```

---

state_labels *Specify a "labels" state*

Description

Specify a "labels" state

Usage

```
state_labels(varnames = NULL)
```

Arguments

varnames	A vector of variable names whose values should appear as labels. If NULL, no labels will be displayed. when viewing the display.

---

state_layout *Specify a "layout" state*

Description

Specify a "layout" state

Usage

```
state_layout(
  ncol = 1,
  page = 1,
  sidebar = FALSE,
  viewtype = c("grid", "table"),
  visible_filters = NULL
)
```

Arguments

ncol	Number of columns of panels to show.
page	The page number to show.
sidebar	Should the sidebar be shown?
viewtype	The type of view to show ("grid" or "table").
visible_filters	A vector of variable names to add as visible filters in the sidebar.

---

state_sort *Specify a "sort" state*

Description

Specify a "sort" state

Usage

```
state_sort(varname, dir = "asc")
```

Arguments

varname	The name of the variable to sort on.
dir	One of "asc" or "desc", describing the direction of the sort.

---

trelliscope-shiny *Shiny bindings for trelliscope*

Description

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

Usage

```
trelliscopeOutput(outputId, ...)

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

Arguments

outputId	output variable to read from
...	Argumentsed passed on to shiny::htmlOutput() is useful if you want to save an expression in a variable.
expr	An expression that generates a trelliscope
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This

update_display_list *Update the list of all displays in an app directory*

Description

Update the list of all displays in an app directory

Usage

```
update_display_list(app_path, jsonp = TRUE, id)
```

Arguments

app_path	The path where all of the displays are stored
jsonp	If true, files are read and written as "jsonp" format, otherwise "json" format. The "jsonp" format makes it possible to browse a trelliscope app without the need for a web server.
id	The id of the display. Can be found in `config.json[p]`.

view_trelliscope *View a trelliscope display*

Description

View a trelliscope display

Usage

```
view_trelliscope(trdf = NULL)
```

Arguments

trdf	A trelliscope data frame created with `as_trelliscope_df()` or a data frame which will be cast as such.

Examples

```
library(ggplot2)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
) |>
  as_panels_df()
```

```
disp <- panel_dat |>
  as_trelliscope_df(name = "life_expectancy")

## Not run:
view_trelliscope(disp)

## End(Not run)
```

`write_trelliscope` *Write the contents of a display*

Description

Write the contents of a display

Usage

```
write_trelliscope(trdf, force_write = NULL, jsonp = NULL)
```

Arguments

`trdf`	A trelliscope data frame created with `as_trelliscope_df()` or a data frame which will be cast as such.
`force_write`	Should the panels be forced to be written even if they have already been written?
`jsonp`	If true, app files are written as "jsonp" format, otherwise "json" format. The "jsonp" format makes it possible to browse a trelliscope app without the need for a web server.

Examples

```
library(ggplot2)

panel_dat <- (
  ggplot(gap, aes(year, life_exp)) +
    geom_point() +
    facet_panels(vars(country, continent))
) |>
  as_panels_df()

disp <- panel_dat |>
  as_trelliscope_df(name = "life_expectancy")

## Not run:
disp <- write_trelliscope(disp)
view_trelliscope(disp)

## End(Not run)
```

Index