

Package: sap (via r-universe)

October 30, 2026

Title Shiny App-Packages

Version 0.0.0.9200

Author John Smith <John.Smith@email.io> [aut, cre]

Maintainer John Smith <John.Smith@email.io>

Description An R package with a collection of Shiny applications.

License GPL-3

Depends R (>= 2.10)

Imports bslib, cli, ggplot2, glue, htmltools, logger, quarto, rlang,
sass, shiny, stringr, tools

Suggests ggplot2movies, knitr, rmarkdown, shinytest2, testthat (>= 3.0.0), waldo

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.1.1

Config/pak/sysreqs cmake make libicu-dev libuv1-dev zlib1g-dev

Repository <https://mjfrigaard.r-universe.dev>

Date/Publication 2026-10-30 16:40:16 UTC

RemoteUrl <https://github.com/mjfrigaard/sap>

RemoteRef 22_webr

RemoteSha 8dfe7b71c42707d6a289a8ad1e13f3bbf88d1208

Contents

display_type	2
launch_app	3
log_message	4

logr_msg	5
mod_aes_input_server	6
mod_aes_input_ui	7
mod_scatter_display_server	8
mod_scatter_display_ui	9
mod_var_input_server	10
mod_var_input_ui	11
movies	12
movies_server	14
movies_ui	15
scatter_plot	16
text_logo	17
Index	18

display_type	<i>Shiny app display mode helper</i>
--------------	--------------------------------------

Description

Shiny app display mode helper

Usage

display_type(run = "w")

Arguments

- | | |
|-----|----------------------|
| run | where to launch app: |
|-----|----------------------|
- "p" = launch in viewer pane
 - "b" = launch in external browser
 - "w" = launch in window (default)

Value

notification of shinyViewerType option

launch_app*Launch the Movies Review Application*

Description

Starts the Movies Review Shiny application, which provides a customizable scatter plot interface for analyzing movie data.

Usage

```
launch_app(app = NULL, options = list(), mode = "run", run = "p", ...)
```

Arguments

app	which app to run. Options are: <ul style="list-style-type: none">• NULL = the default app ("movies")• "bslib" = alternative bslib layout• "ggp2" = ggplot2movies (tidy) data app.• "quarto" = Quarto movies app.
options	arguments to pass to options()
mode	launch mode. Choices are: <ul style="list-style-type: none">• run = blocking, uses <code>display_type()</code> for window placement (default)• start = non-blocking, launches in a background process
run	where to launch app: <ul style="list-style-type: none">• p = launch in viewer pane• b = launch in external browser• w = launch in window
...	arguments passed to UI

Details

See the `ggplot2movies` package.

Value

A **Shiny application** object.

Details

The application uses:

- **UI:** Defined in `movies_ui()`.
- **Server Logic:** Defined in `movies_server()`.

See Also

- [movies_ui\(\)](#) for the user interface.
- [movies_server\(\)](#) for the server logic.

Examples

```
if (interactive()) {  
  launch_app()  
}
```

log_message

Log Application Behaviors

Description

A simple logging function for logging Shiny events and behaviors.

Usage

```
log_message(message, log_file = "logs/app_log.txt", save = FALSE)
```

Arguments

message	log message
log_file	name of log file (defaults to "app_log.txt")
save	print or save log file

Value

A log message to the console and the log_file.

See Also

Other **Utility Functions**: [scatter_plot\(\)](#)

Examples

```
if (interactive()) {  
  log_message("message")  
}
```

logr_msg

Log Messages to Console and File with Optional JSON Format

Description

A utility function for logging messages to the console and optionally to a log file. Messages can be stored as plain text (default) or in JSON format. Additionally, logging to a file can be disabled altogether.

Usage

```
logr_msg(  
    message,  
    level = "INFO",  
    log_file = "app_log.txt",  
    json = FALSE,  
    store_log = FALSE  
)
```

Arguments

message	A character string representing the message to be logged.
level	A character string indicating the log level. One of: <ul style="list-style-type: none">• "FATAL"• "ERROR"• "WARN"• "SUCCESS"• "INFO" (default)• "DEBUG"• "TRACE"
log_file	A character string specifying the path to the log file. Ignored if store_log = FALSE. Defaults to "app_log.txt".
json	A logical value indicating whether to save logs in JSON format. Defaults to FALSE. When TRUE, logs are written to log_file as JSON objects.
store_log	A logical value indicating whether logs should be saved to a file. Defaults to TRUE: <ul style="list-style-type: none">• If TRUE, logs are printed to the console and stored in log_file• If FALSE, logs are printed to the console

Details

- **Console Logging:** All messages are always logged to the console.
- **File Logging:** If store_log = TRUE and log_file is specified, messages are logged to the file in either plain text or JSON format.

- **JSON Logs:** When `json = TRUE`, log messages are stored as structured JSON objects with the fields:
 - `timestamp`: The time the message was logged.
 - `level`: The severity level of the log.
 - `message`: The actual log message.

Value

The function is called for its side effects of logging messages and does not return a value.

See Also

- `logger::log_appender()` for configuring log destinations.
- `logger::log_formatter()` for customizing log message formats.
- `jsonlite::toJSON()` for converting R objects to JSON.

Examples

```
# Log a simple informational message to the console
logr_msg("Application started.")

# Log an error message to the console and a file
logr_msg("Failed to connect to database.", level = "ERROR", log_file = "error_log.txt")

# Log a success message in JSON format
logr_msg("User login successful.", level = "SUCCESS", log_file = "app_log.json", json = TRUE)

# Log a warning message to the console only (no file storage)
logr_msg("Low memory warning.", level = "WARN", store_log = FALSE)
```

mod_aes_input_server *Aesthetics Input Module - Server*

Description

Handles the server-side logic for the variable input module.

Usage

```
mod_aes_input_server(id)
```

Arguments

`id` (*character*) Namespace ID for the module.

Value

A **reactive expression** that returns a list of selected input values:

- alpha: Transparency level of points (0-1).
- size: Size of the points.
- plot_title: Title for the scatter plot.

Details

mod_aes_input_server() reads user input from the corresponding UI function created with mod_aes_input_ui(). It processes and returns a reactive object containing the selected plot attributes.

See Also

- [mod_aes_input_ui\(\)](#) for the UI counterpart of this module.
- [mod_scatter_display_server\(\)](#) for displaying the scatter plot.

mod_aes_input_ui	<i>Aesthetics Input Module - UI</i>
------------------	-------------------------------------

Description

Creates a UI for selecting attributes for a scatter plot.

Usage

```
mod_aes_input_ui(id)
```

Arguments

id *(character)* Namespace ID for the module.

Value

A shiny::tagList() containing UI elements for variable selection.

Details

mod_aes_input_ui() provides UI controls for customizing scatter plots. It includes:

- **Sliders:**
 - Transparency (alpha) ranging from 0 to 1.
 - Point size ranging from 0 to 5.
- **Text Input:**
 - Plot title.

The function uses shiny::NS() to namespace all input IDs.

See Also

- `mod_aes_input_server()` for server-side logic.
- `shiny::tagList()` for containing UI elements.
- `shiny::NS()` for namespacing.

Examples

```
if (interactive()) {
  shiny::shinyApp(
    ui = shiny::fluidPage(mod_aes_input_ui("aes")),
    server = function(input, output, session) {
      selected_vars <- mod_aes_input_server("aes")
      shiny::observe(
        print(selected_vars())
      )
    }
  )
}
```

mod_scatter_display_server

Scatter Plot Display Module - Server

Description

Handles the server-side logic for rendering a scatter plot.

Usage

```
mod_scatter_display_server(id, var_inputs, aes_inputs)
```

Arguments

<code>id</code>	<i>(character)</i> Namespace ID for the module.
<code>var_inputs</code>	<i>(reactive)</i> A reactive expression containing user-selected variables.
<code>aes_inputs</code>	<i>(reactive)</i> A reactive expression containing user-selected attributes.

Value

No direct return value. This function generates a plot output.

Details

```
mod_scatter_display_server():
```

- Uses `var_inputs` to dynamically generate a scatter plot with user-selected variables.
- Reads from the `movies` dataset, which must be loaded in the app environment.
- Processes plot titles and axis labels to improve readability.

Reactive Inputs

- `var_inputs()`\$x: X-axis variable.
- `var_inputs()`\$y: Y-axis variable.
- `var_inputs()`\$z: Color aesthetic variable.
- `aes_inputs()`\$alpha: Transparency level.
- `aes_inputs()`\$size: Size of points.
- `aes_inputs()`\$plot_title: Title of the plot.

See Also

- [mod_var_input_server\(\)](#) for variable selection.
- [mod_aes_input_server\(\)](#) for aesthetics selection.
- [scatter_plot\(\)](#) for generating the scatter plot.

Other **Plot Display Module**: [mod_scatter_display_ui\(\)](#)

Examples

```
if (interactive()) {  
  shiny::shinyApp(  
    ui = shiny::fluidPage(  
      mod_var_input_ui("vars"),  
      mod_aes_input_ui("aes"),  
      mod_scatter_display_ui("plot")  
    ),  
    server = function(input, output, session) {  
      selected_vars <- mod_var_input_server("vars")  
      selected_aes <- mod_aes_input_server("aes")  
      mod_scatter_display_server("plot", selected_vars, selected_aes)  
    }  
  )  
}
```

mod_scatter_display_ui

Scatter Plot Display Module - UI

Description

Creates a UI for displaying a scatter plot with user-selected options.

Usage

```
mod_scatter_display_ui(id)
```

Arguments

`id` *(character)* Namespace ID for the module.

Value

A `shiny::tagList` containing the plot output and metadata.

Details

`mod_scatter_display_ui()` includes:

- A **scatter plot** created dynamically based on user input.
- Metadata about the app, including the data source.

See Also

- [mod_scatter_display_server\(\)](#) for the server-side logic of this module.
- [scatter_plot\(\)](#) for the utility function generating the plot.

Other **Plot Display Module**: [mod_scatter_display_server\(\)](#)

Examples

```
if (interactive()) {  
  shiny::shinyApp(  
    ui = shiny::fluidPage(mod_scatter_display_ui("plot")),  
    server = function(input, output, session) {  
      # Example usage - server logic is minimal for standalone testing.  
    }  
  )  
}
```

mod_var_input_server *Variable Input Module - Server*

Description

Handles the server-side logic for the variable input module.

Usage

```
mod_var_input_server(id)
```

Arguments

`id` *(character)* Namespace ID for the module.

Value

A **reactive expression** that returns a list of selected input values:

- y: Variable for the y-axis.
- x: Variable for the x-axis.
- z: Variable for the point colors.

Details

mod_var_input_server() reads user input from the corresponding UI function created with mod_var_input_ui(). It processes and returns a reactive object containing the selected variables.

See Also

- [mod_var_input_ui\(\)](#) for the UI counterpart of this module.
- [mod_scatter_display_server\(\)](#) for displaying the scatter plot.

Other **Variable Input Module**: [mod_var_input_ui\(\)](#)

Examples

```
if (interactive()) {  
  shiny::shinyApp(  
    ui = shiny::fluidPage(mod_var_input_ui("vars")),  
    server = function(input, output, session) {  
      selected_vars <- mod_var_input_server("vars")  
      shiny::observe(print(selected_vars()))  
    }  
  )  
}
```

mod_var_input_ui	<i>Variable Input Module - UI</i>
------------------	-----------------------------------

Description

Creates a UI for selecting variables for a scatter plot.

Usage

```
mod_var_input_ui(id)
```

Arguments

id *(character)* Namespace ID for the module.

Value

A `shiny::tagList()` containing UI elements for variable selection.

Details

`mod_var_input_ui()` provides UI controls for customizing scatter plots. It includes:

- **Dropdowns:**
 - X-axis variable
 - Y-axis variable
 - Color variable

The function uses `shiny::NS()` to namespace all input IDs.

See Also

- `mod_var_input_server()` for server-side logic.
- `shiny::tagList()` for containing UI elements.
- `shiny::NS()` for namespacing.

Other **Variable Input Module**: `mod_var_input_server()`

Examples

```
if (interactive()) {
  shiny::shinyApp(
    ui = shiny::fluidPage(mod_var_input_ui("vars")),
    server = function(input, output, session) {
      selected_vars <- mod_var_input_server("vars")
      shiny::observe(
        print(selected_vars())
      )
    }
  )
}
```

movies

IMDB movies data

Description

Movie review data. Note: these data come from the [Building Web Applications with shiny course](#).

Usage

```
movies
```

Format

A data frame with 651 rows and 34 variables:

title movie title
title_type type, fct (Documentary, Feature Film, TV Movie)
genre movie genre, fct (Action & Adventure, Animation, etc.)
runtime movie length in minutes, num, avg = 106, sd = 19.4
mpaa_rating movie rating, fct (G, NC-17, PG, PG-13, R, Unrated)
studio name of studio, chr
thtr_rel_date Theatre release date, POSIXct, min = 1970-05-19 21:00:00, max = 2014-12-24 21:00:00
thtr_rel_year Theatre release year, num, min = 1970, max = 2014
thtr_rel_month Theatre release month, num, min = 1, max = 12
thtr_rel_day Theatre release day, num, min = 1, max = 31
dvd_rel_date DVD release date, POSIXct, min = 1991-03-27 21:00:00, max = 2015-03-02 21:00:00
dvd_rel_year DVD release year, num, min = 1991, max = 2015
dvd_rel_month DVD release month, num, min = 1, max = 12
dvd_rel_day DVD release day, num, min = 1, max = 31
imdb_rating Internet movie database rating, avg = 6.49, sd = 1.08
imdb_num_votes Internet movie database votes, avg = 57533, sd = 112124
critics_rating Rotten tomatoes rating, fct (Certified Fresh, Fresh, Rotten)
critics_score Rotten tomatoes score, avg = 57.7, sd = 28.4
audience_rating Audience rating, fct (Spilled, Upright)
audience_score Audience score, avg = 62.4, sd = 20.2
best_pic_nom Best picture nomination, fct (no, yes)
best_pic_win Best picture win, fct (no, yes)
best_actor_win Best actor win, fct (no, yes)
best_actress_win Best actress win, fct (no, yes)
best_dir_win Best director win, fct (no, yes)
top200_box Top 20 box-office, fct (no, yes)
director Name of director, chr
actor1 Name of leading actor, chr
actor2 Name of supporting actor, chr
actor3 Name of #3 actor, chr
actor4 Name of #4 actor, chr
actor5 Name of #5 actor, chr
imdb_url IMDB URL
rt_url Rotten tomatoes URL

Details

Read more about acquiring these data in the ['Web Scraping and programming'](#) section of [Data science in a box](#)

`movies_server`*Handles the server-side logic for the Movies Review application.*

Description

Handles the server-side logic for the Movies Review application.

Usage

```
movies_server(input, output, session)
```

Arguments

<code>input</code>	The Shiny input object.
<code>output</code>	The Shiny output object.
<code>session</code>	The Shiny session object.

Value

Defines server logic for reactive outputs and module integration.

Details

The server integrates the following modules:

- **Variable Input Module:** Processes user input for scatter plot customization using `mod_var_input_server()`.
- **Plot Display Module:** Generates and displays a scatter plot based on user inputs using `mod_scatter_display_server()`.

See Also

- `movies_ui()` for the corresponding user interface.
- `launch_app()` for the entry point of the application.

Other **Application Components**: `movies_ui()`

Examples

```
if (interactive()) {  
  shiny::shinyApp(ui = movies_ui(), server = movies_server)  
}
```

Description

Creates the user interface (UI) for the Movies Review application, which allows users to create customizable scatter plots based on movie data.

Usage

```
movies_ui(bslib = FALSE)
```

Arguments

bslib	View bslib logo?
-------	------------------

Value

A Shiny tagList object containing the UI elements.

Details

The interface is built using **bslib**

- **Page (fillable):** `bslib::page_fillable()` displays the app title.
- **Sidebar:** `bslib::layout_sidebar()` includes a logo and the variable selection module (`mod_var_input_ui`).
- **Card:** `bslib::card()` displays the scatter plot module (`mod_scatter_display_ui`).

See Also

- `movies_server()` for the server logic of the app.
- `mod_var_input_ui()` and `mod_scatter_display_ui()` for the modules included in the UI.

Other **Application Components**: `movies_server()`

Examples

```
if (interactive()) {  
  shiny::shinyApp(ui = movies_ui(), server = movies_server)  
}
```

scatter_plot	<i>Generate a Scatter Plot</i>
--------------	--------------------------------

Description

Creates a scatter plot using ggplot2 with the specified data and aesthetics.

Usage

```
scatter_plot(df, x_var, y_var, col_var, alpha_var, size_var)
```

Arguments

df	(<i>data.frame</i>) The dataset containing the variables to plot.
x_var	(<i>character</i>) Name of the variable for the x-axis.
y_var	(<i>character</i>) Name of the variable for the y-axis.
col_var	(<i>character</i>) Name of the variable for the color aesthetic.
alpha_var	(<i>numeric</i>) Transparency level of points (0 to 1).
size_var	(<i>numeric</i>) Size of points.

Value

A ggplot object representing the scatter plot.

Details

scatter_plot() is designed for use in Shiny applications but can also be used independently. It supports customization of transparency, size, and color aesthetics.

See Also

- [mod_scatter_display_server\(\)](#) for integrating this function into the scatter plot module.
- [ggplot2::ggplot\(\)](#) for details on ggplot2 usage.

Other **Utility Functions**: [log_message\(\)](#)

Examples

```
scatter_plot(  
  df = mtcars,  
  x_var = "mpg",  
  y_var = "hp",  
  col_var = "cyl",  
  alpha_var = 0.7,  
  size_var = 3  
)
```

`text_logo`*Create a Text-based Logo in HTML*

Description

`text_logo()` generates a text-based logo enclosed within HTML tags, allowing for the specification of the heading size (h1 through h6). The logo features a stylized representation of a face within arrows, using ASCII characters.

Usage

```
text_logo(size = "h3")
```

Arguments

<code>size</code>	A character string specifying the heading size for the logo. Valid options include "h1", "h2", "h3", "h4", "h5", and "h6". Defaults to "h3".
-------------------	--

Value

An HTML object containing the logo. This object can be directly used in R Markdown documents or Shiny applications to render the logo in a web page.

Examples

```
# Generate a logo with default size h3
text_logo()
```

```
# Generate a logo with size h1
text_logo(size = "h1")
```

Index

- * **datasets**
 - movies, [12](#)
- * **if (interactive())**
 - mod_aes_input_server, [6](#)
- * **list(Aesthetics Input Module)**
 - mod_aes_input_server, [6](#)
 - mod_aes_input_ui, [7](#)
- * **list(Application Components)**
 - movies_server, [14](#)
 - movies_ui, [15](#)
- * **list(Plot Display Module)**
 - mod_scatter_display_server, [8](#)
 - mod_scatter_display_ui, [9](#)
- * **list(Standalone Application)**
 - launch_app, [3](#)
- * **list(Utility Functions)**
 - log_message, [4](#)
 - scatter_plot, [16](#)
- * **list(Variable Input Module)**
 - mod_var_input_server, [10](#)
 - mod_var_input_ui, [11](#)
- * **list(\n, shiny::shinyApp(\n, ui =**
 - shiny::fluidPage(mod_var_input_ui(\nvars)),\n,**
 - server = function(input, output,**
 - session) , list(\n, selected_vars <-**
 - mod_var_input_server(\nvars))\n,**
 - shiny::observe(print(selected_vars()))\n),**
 - \n,)\n)**
 - mod_aes_input_server, [6](#)
- *
- mod_aes_input_server, [6](#)
- display_type, [2](#)
- display_type(), [3](#)
- jsonlite::toJSON(), [6](#)
- launch_app, [3](#)
- launch_app(), [14](#)
- log_message, [4](#)
- log_message(), [16](#)
- logger::log_appender(), [6](#)
- logger::log_formatter(), [6](#)
- logr_msg, [5](#)
- mod_aes_input_server, [6](#)
- mod_aes_input_server(), [8](#), [9](#)
- mod_aes_input_ui, [7](#)
- mod_aes_input_ui(), [7](#)
- mod_scatter_display_server, [8](#)
- mod_scatter_display_server(), [7](#), [10](#), [11](#), [14](#), [16](#)
- mod_scatter_display_ui, [9](#), [15](#)
- mod_scatter_display_ui(), [9](#), [15](#)
- mod_var_input_server, [10](#)
- mod_var_input_server(), [9](#), [12](#), [14](#)
- mod_var_input_ui, [11](#), [15](#)
- mod_var_input_ui(), [11](#), [15](#)
- movies, [12](#)
- movies_server, [14](#)
- movies_server(), [3](#), [4](#), [15](#)
- movies_ui, [15](#)
- movies_ui(), [3](#), [4](#), [14](#)
- scatter_plot, [16](#)
- scatter_plot(), [4](#), [9](#), [10](#)
- text_logo, [17](#)