

Package ‘maidr’

April 1, 2026

Type Package

Title Multimodal Access and Interactive Data Representation

Version 0.3.0

URL <https://github.com/xability/r-maidr>, <https://r.maidr.ai/>

BugReports <https://github.com/xability/r-maidr/issues>

Description Provides accessible, interactive visualizations through the 'MAIDR' (Multimodal Access and Interactive Data Representation) system. Converts 'ggplot2' and Base R plots into accessible HTML/SVG formats with keyboard navigation, screen reader support, and 'sonification' capabilities. Supports bar charts (simple, grouped, stacked), histograms, line plots, scatter plots, box plots, violin plots, heat maps, density/smooth curves, faceted plots, multi-panel layouts (including patchwork), and multi-layered plot combinations. Enables data exploration for users with visual impairments through multiple sensory modalities. For more details see the 'MAIDR' project <<https://maidr.ai/>>.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 3.5.0)

Imports base64enc, curl, ggplot2, ggplotify, grid, gridSVG, htmltools, htmlwidgets, jsonlite, shiny, xml2, rlang, R6

Suggests testthat (>= 3.0.0), lintr, styler, goodpractice, cyclocomp, knitr, rmarkdown, quarto, patchwork

Config/testthat/edition 3

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation no

Author JooYoung Seo [aut, cph],
Niranjan Kalaiselvan [aut, cre]

Maintainer Niranjan Kalaiselvan <nk46@illinois.edu>

Repository CRAN

Date/Publication 2026-04-01 05:30:02 UTC

Contents

combine_facet_layer_data	2
combine_facet_layer_selectors	3
extract_leaf_plot_layout	3
extract_patchwork_leaves	4
find_children_by_type	4
find_graphics_plot_grob	5
find_panel_grob	5
find_patchwork_panels	6
generate_robust_css_selector	6
generate_robust_selector	7
get_facet_groups	7
maidr_get_fallback	8
maidr_off	9
maidr_on	9
maidr_output	10
maidr_set_fallback	11
map_visual_to_dom_panel	12
organize_facet_grid	12
process_facet_panel	13
process_patchwork_panel	14
render_maidr	14
run_example	15
save_html	16
show	17
Index	19

combine_facet_layer_data

Combine data from multiple layers in facet processing

Description

Combine data from multiple layers in facet processing

Usage

```
combine_facet_layer_data(layer_results)
```

Arguments

layer_results List of layer processing results

Value

Combined data

`combine_facet_layer_selectors`

Combine selectors from multiple layers in facet processing

Description

Combine selectors from multiple layers in facet processing

Usage

```
combine_facet_layer_selectors(layer_results)
```

Arguments

`layer_results` List of layer processing results

Value

Combined selectors

`extract_leaf_plot_layout`

Extract layout from a single leaf ggplot

Description

Extract layout from a single leaf ggplot

Usage

```
extract_leaf_plot_layout(leaf_plot)
```

Arguments

`leaf_plot` The ggplot object

Value

Layout with title and axes

extract_patchwork_leaves

Recursively extract leaf ggplots in visual order

Description

Recursively extract leaf ggplots in visual order

Usage

```
extract_patchwork_leaves(node)
```

Arguments

node Patchwork node or ggplot object

Value

List of leaf ggplot objects

find_children_by_type *Find children matching a type pattern*

Description

Find children matching a type pattern

Usage

```
find_children_by_type(parent_grob, pattern)
```

Arguments

parent_grob The parent grob to search
pattern The pattern to match in grob names

Value

Vector of matching child names

`find_graphics_plot_grob`*Utility functions for robust selector generation in Base R plots*

Description

These functions provide a robust way to find grob elements and generate CSS selectors, independent of panel structure or hardcoded values. Find grob by element type pattern

Usage

```
find_graphics_plot_grob(grob, element_type, plot_index = NULL)
```

Arguments

<code>grob</code>	The grob tree to search (typically from <code>ggplotify::as.grob()</code>)
<code>element_type</code>	The element type to search for (e.g., "rect", "lines", "points")
<code>plot_index</code>	Optional plot index to match (for multipanel layouts)

Details

Searches recursively through a grob tree to find a grob whose name matches the pattern: `graphics-plot-<number>-<element_type>-<number>`

Value

The name of the first matching grob, or NULL if not found

`find_panel_grob`*Find the panel grob in a grob tree*

Description

Find the panel grob in a grob tree

Usage

```
find_panel_grob(grob_tree)
```

Arguments

<code>grob_tree</code>	The grob tree to search
------------------------	-------------------------

Value

The panel grob or NULL if not found

`find_patchwork_panels` *Discover panels via gtable layout rows named '^panel-<num>' or '^panel-<row>-<col>' Returns a data.frame with panel_index, name, t, l, row, col*

Description

Discover panels via gtable layout rows named '^panel-<num>' or '^panel-<row>-<col>' Returns a data.frame with panel_index, name, t, l, row, col

Usage

```
find_patchwork_panels(gtable)
```

Arguments

`gtable` Gtable object

Value

Data frame with panel information

`generate_robust_css_selector`
Generate robust CSS selector from grob name

Description

Creates a CSS selector that targets SVG elements by their ID pattern, without relying on panel structure or hardcoded values.

Usage

```
generate_robust_css_selector(grob_name, svg_element)
```

Arguments

`grob_name` The name of the grob (e.g., "graphics-plot-1-rect-1")
`svg_element` The SVG element type to target (e.g., "rect", "polyline")

Value

A robust CSS selector string, or NULL if grob_name is invalid

`generate_robust_selector`*Generate robust selector for any element type*

Description

Creates a robust CSS selector that works regardless of panel structure. This is the main function that layer processors should use.

Usage

```
generate_robust_selector(  
  grob,  
  element_type,  
  svg_element,  
  plot_index = NULL,  
  max_elements = NULL  
)
```

Arguments

<code>grob</code>	The grob tree to analyze
<code>element_type</code>	The element type to search for (e.g., "rect", "lines")
<code>svg_element</code>	The SVG element to target (e.g., "rect", "polyline")
<code>plot_index</code>	Optional plot index for multipanel layouts
<code>max_elements</code>	Optional limit on number of elements to target

Value

A robust CSS selector string, or NULL if element not found

`get_facet_groups`*Get facet group information for a panel*

Description

Get facet group information for a panel

Usage

```
get_facet_groups(panel_info, built)
```

Arguments

panel_info	Panel information from layout
built	Built plot data

Value

List of facet group information

maidr_get_fallback *Get Current MAIDR Fallback Settings*

Description

Retrieves the current fallback configuration for MAIDR.

Usage

```
maidr_get_fallback()
```

Value

A list with the current fallback settings:

- enabled: Logical indicating if fallback is enabled
- format: Character string of the image format
- warning: Logical indicating if warnings are shown

See Also

[maidr_set_fallback()] to configure settings

Examples

```
# Get current settings
settings <- maidr_get_fallback()
print(settings)
```

`maidr_off`*Disable MAIDR Plot Interception*

Description

Disables automatic MAIDR rendering and restores normal plot behavior. After calling this, Base R plots display in the standard graphics window and ggplot2 objects render with the default ggplot2 method.

Usage

```
maidr_off()
```

Value

Invisible TRUE on success

See Also

[`maidr_on()`] to enable MAIDR rendering

`maidr_on`*Enable MAIDR Plot Interception*

Description

Enables automatic accessible rendering of ggplot2 and Base R plots. In interactive sessions, plots are displayed in the MAIDR interactive viewer. In RMarkdown documents, plots are converted to accessible MAIDR widgets with keyboard navigation and screen reader support.

Usage

```
maidr_on()
```

Value

Invisible TRUE on success

See Also

[`maidr_off()`] to disable MAIDR rendering

Examples

```
library(maidr)

# Enable interception (on by default after library(maidr))
maidr_on()

# Now all plots render as accessible MAIDR widgets
library(ggplot2)
ggplot(mtcars, aes(x = factor(cyl))) +
  geom_bar()

barplot(table(mtcars$cyl))
```

maidr_output

MAIDR Output Container for Shiny UI

Description

Creates a Shiny output container for MAIDR widgets using `htmlwidgets`. This provides automatic dependency injection and robust JavaScript initialization.

Usage

```
maidr_output(output_id, width = "100%", height = "400px")
```

Arguments

<code>output_id</code>	The output variable to read the plot from
<code>width</code>	The width of the plot container (default: "100percent")
<code>height</code>	The height of the plot container (default: "400px")

Value

A Shiny widget output function for use in UI

Examples

```
if (interactive()) {
  library(shiny)
  ui <- fluidPage(maidr_output("myplot"))
}
```

maidr_set_fallback *Configure MAIDR Fallback Behavior*

Description

Configure how MAIDR handles unsupported plot types or layers. When fallback is enabled, unsupported plots are rendered as static images instead of failing or returning empty data.

Usage

```
maidr_set_fallback(enabled = TRUE, format = "png", warning = TRUE)
```

Arguments

enabled	Logical. If TRUE (default), unsupported plots fall back to image rendering. If FALSE, unsupported layers return empty data.
format	Character. Image format for fallback: "png" (default), "svg", or "jpeg".
warning	Logical. If TRUE (default), shows a warning message when falling back to image rendering.

Value

Invisibly returns a list of the previous settings.

See Also

[maidr_get_fallback()] to retrieve current settings

Examples

```
# Save current settings and restore on exit
old_settings <- maidr_get_fallback()
on.exit(maidr_set_fallback(
  enabled = old_settings$enabled,
  format = old_settings$format,
  warning = old_settings$warning
))

# Disable fallback (unsupported plots will have empty data)
maidr_set_fallback(enabled = FALSE)

# Use SVG format for fallback images
maidr_set_fallback(format = "svg")

# Disable warning messages
maidr_set_fallback(warning = FALSE)

# Configure multiple options
```

```
maidr_set_fallback(enabled = TRUE, format = "png", warning = TRUE)
```

```
map_visual_to_dom_panel
```

Map visual panel position to DOM panel name

Description

This function handles the mismatch between visual layout order (row-major) and DOM element generation order (column-major) in gridSVG.

Usage

```
map_visual_to_dom_panel(panel_info, gtable)
```

Arguments

panel_info	Panel information from layout
gtable	Gtable object

Details

Visual layout (row-major): 1 2 3 4
 DOM order (column-major): 1 3 2 4

Value

Gtable panel name or NULL if not found

```
organize_facet_grid
```

Organize subplots into 2D grid structure

Description

Organize subplots into 2D grid structure

Usage

```
organize_facet_grid(subplots, panel_layout)
```

Arguments

subplots	List of processed subplot data
panel_layout	Panel layout information

Value

2D grid structure

process_facet_panel *Process a single facet panel*

Description

Process a single facet panel

Usage

```
process_facet_panel(
  plot,
  panel_info,
  panel_data,
  facet_groups,
  gtable_panel_name,
  built,
  layout,
  gtable,
  format_config = NULL
)
```

Arguments

plot	The original plot
panel_info	Panel information
panel_data	Panel-specific data
facet_groups	Facet group information
gtable_panel_name	Gtable panel name
built	Built plot data
layout	Layout information
gtable	Gtable object
format_config	Optional format configuration from maird label functions

Value

Processed panel data

```
process_patchwork_panel
```

Process a single patchwork panel

Description

Process a single patchwork panel

Usage

```
process_patchwork_panel(  
  leaf_plot,  
  panel_name,  
  panel_index,  
  row,  
  col,  
  layout,  
  gtable  
)
```

Arguments

leaf_plot	The leaf ggplot object
panel_name	Panel name from gtable
panel_index	Panel index
row	Panel row
col	Panel column
layout	Layout information
gtable	Gtable object

Value

Processed panel data

```
render_maidr
```

Render MAIDR Plot in Shiny Server

Description

Creates a Shiny render function for MAIDR widgets using htmlwidgets. This provides automatic dependency injection and robust JavaScript initialization.

Usage

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

Arguments

expr	An expression that returns a ggplot object
env	The environment in which to evaluate expr
quoted	Is expr a quoted expression

Value

A Shiny render function for use in server

Examples

```
if (interactive()) {
  library(shiny)
  library(ggplot2)
  server <- function(input, output) {
    output$myplot <- render_maidr({
      ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
        geom_bar(stat = "identity")
    })
  }
}
```

run_example

Run MAIDR Example Plots

Description

Launches example plots demonstrating MAIDR's accessible visualization capabilities. Each example creates an interactive plot using 'show()'.

Usage

```
run_example(example = NULL, type = c("ggplot2", "base_r"))
```

Arguments

example	Character string specifying which example to run. If 'NULL' (the default), lists all available examples.
type	Character string specifying the plot system to use. Either "ggplot2" (default) or "base_r".

Details

Available examples include various plot types such as bar charts, histograms, scatter plots, line plots, boxplots, heatmaps, and more.

Each example script creates a plot and calls ‘show()’ to display it in your default web browser with full MAIDR accessibility features including keyboard navigation and screen reader support.

Value

Invisibly returns ‘NULL’. Called for its side effect of displaying an interactive plot in the browser or listing available examples.

See Also

[show()] for displaying plots, [save_html()] for saving to file

Examples

```
# List all available examples
run_example()

if (interactive()) {
  # Run ggplot2 bar chart example
  run_example("bar")

  # Run Base R histogram example
  run_example("histogram", type = "base_r")
}
```

save_html

Save Interactive Plot as HTML File

Description

Save a ggplot2 or Base R plot as a standalone HTML file with interactive MAIDR accessibility features.

Usage

```
save_html(plot = NULL, file = "plot.html", ...)
```

Arguments

plot	A ggplot2 object or NULL for Base R auto-detection
file	File path where to save the HTML file (e.g., "plot.html")
...	Additional arguments passed to internal functions

Value

The file path where the HTML was saved (invisibly)

Examples

```
# ggplot2 bar chart
library(ggplot2)
p <- ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_bar(stat = "identity")

maidr::save_html(p, tempfile(fileext = ".html"))

# ggplot2 violin plot
p_violin <- ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_violin(fill = "lightblue", alpha = 0.7) +
  labs(title = "MPG by Cylinders", x = "Cylinders", y = "MPG")

maidr::save_html(p_violin, tempfile(fileext = ".html"))

# Base R example (requires interactive session for function patching)
if (interactive()) {
  barplot(c(10, 20, 30), names.arg = c("A", "B", "C"))
  maidr::save_html(file = tempfile(fileext = ".html"))
}
```

 show

Display Interactive MAIDR Plot

Description

Display a ggplot2 or Base R plot as an interactive, accessible visualization using the MAIDR (Multimodal Access and Interactive Data Representation) system.

Usage

```
show(plot = NULL, shiny = FALSE, as_widget = FALSE, ...)
```

Arguments

plot	A ggplot2 object or NULL for Base R auto-detection
shiny	If TRUE, returns just the SVG content instead of full HTML document
as_widget	If TRUE, returns an htmlwidget object instead of opening in browser
...	Additional arguments passed to internal functions

Value

Invisible NULL. The plot is displayed in RStudio Viewer or browser as a side effect.

Examples

```
# ggplot2 bar chart
library(ggplot2)
p <- ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_bar(stat = "identity")

maidr::show(p)

# ggplot2 violin plot
p_violin <- ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_violin(fill = "lightblue", alpha = 0.7) +
  labs(title = "MPG by Cylinders", x = "Cylinders", y = "MPG")

maidr::show(p_violin)

# Base R example (requires interactive session for function patching)
if (interactive()) {
  barplot(c(10, 20, 30), names.arg = c("A", "B", "C"))
  maidr::show()
}
```

Index

[combine_facet_layer_data](#), 2
[combine_facet_layer_selectors](#), 3

[extract_leaf_plot_layout](#), 3
[extract_patchwork_leaves](#), 4

[find_children_by_type](#), 4
[find_graphics_plot_grob](#), 5
[find_panel_grob](#), 5
[find_patchwork_panels](#), 6

[generate_robust_css_selector](#), 6
[generate_robust_selector](#), 7
[get_facet_groups](#), 7

[maidr_get_fallback](#), 8
[maidr_off](#), 9
[maidr_on](#), 9
[maidr_output](#), 10
[maidr_set_fallback](#), 11
[map_visual_to_dom_panel](#), 12

[organize_facet_grid](#), 12

[process_facet_panel](#), 13
[process_patchwork_panel](#), 14

[render_maidr](#), 14
[run_example](#), 15

[save_html](#), 16
[show](#), 17