Package: dockViewR (via r-universe)

May 12, 2026

Title Layout Manager Widget for R and 'shiny' Apps

Version 0.3.0

Description Provides R bindings to the 'dockview' 'JavaScript' library <https://dockview.dev/>. Create fully customizable grid layouts (docks) in seconds to include in interactive R reports with R Markdown or 'Quarto' or in 'shiny' apps <https://shiny.posit.co/>. In 'shiny' mode, modify docks by dynamically adding, removing or moving panels or groups of panels from the server function. Choose among 8 stunning themes (dark and light), serialise the state of a dock to restore it later.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE, packages = c(` `roxy.shinylive"))

RoxygenNote 7.3.3

Imports htmlwidgets, htmltools, shiny

Suggests knitr, rmarkdown, roxy.shinylive, shinytest2, testthat (>= 3.0.0), visNetwork, quarto, thematic, listviewer, jsonlite, withr

URL https://github.com/cynkra/dockViewR, https://cynkra.github.io/dockViewR/

BugReports https://github.com/cynkra/dockViewR/issues

Config/testthat/edition 3

Depends R (>= 2.10)

VignetteBuilder quarto

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

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

Date/Publication 2026-05-12 07:49:14 UTC

RemoteUrl https://github.com/cynkra/dockViewR

RemoteRef HEAD

RemoteSha 198236ad9264b63528021fa0d7396034a55d2e94

Contents

add_panel 2
default_add_tab_callback 3
default_remove_tab_callback 4
dock_view 4
dock_view_plugins 5
dock_view_proxy 6
dockViewOutput 7
get_dock 7
is_add_tab_plugin 9
is_dock_view_plugin 10
is_remove_tab_plugin 10
panel 11
update_dock_view 12

Index **13**

`add_panel` *Dockview Panel Operations*

Description

Functions to dynamically manipulate panels in a dockview instance.

Usage

```
add_panel(dock, panel, ...)

remove_panel(dock, id)

set_panel_title(dock, id, title)

select_panel(dock, id)

move_panel(dock, id, position = NULL, group = NULL, index = NULL)

move_group(dock, from, to, position = NULL)

move_group2(dock, from, to, position = NULL)
```

Arguments

`dock`	Dock proxy object created with `dock_view_proxy()`.
`panel`	A panel object (for `add_panel`). See panel for parameters.
`...`	Additional options (currently unused).
`id`	Panel ID (character string).

`title`	New panel title.
`position`	Panel/group position: one of "left", "right", "top", "bottom", "center".
`group`	ID of a panel that belongs to the target group (for `move_panel`).
`index`	Panel index within a group (for `move_panel`).
`from`	Source group/panel ID (for move operations).
`to`	Destination group/panel ID (for move operations).

Details

- `set_panel_title()`: Sets the title of a panel dynamically.
- `add_panel()`: Adds a new panel to the dockview
- `remove_panel()`: Removes an existing panel
- `select_panel()`: Selects/focuses a specific panel
- `move_panel()`: Moves a panel to a new position
- `move_group()`: Moves a group using group IDs
- `move_group2()`: Moves a group using panel IDs

Value

All functions return the dock proxy object invisibly, allowing for method chaining.

See Also

`panel()`

`default_add_tab_callback` *Default add tab callback*

Description

An example of a JavaScript function that can be used as a default when adding a new tab/panel.

Usage

```
default_add_tab_callback()
```

Value

An object of class `JS_EVAL` representing the JavaScript callback.

default_remove_tab_callback *Default remove tab callback*

Description

An example of a JavaScript function that can be used as a default when removing a tab/panel.

Usage

```
default_remove_tab_callback()
```

Value

An object of class `JS_EVAL` representing the JavaScript callback.

dock_view *Create a dock view widget*

Description

Creates an interactive dock view widget that enables flexible layout management with draggable, resizable, and dockable panels. This is a wrapper around the dockview.dev JavaScript library, providing a powerful interface for creating IDE-like layouts in Shiny applications or R Markdown documents.

Usage

```
dock_view(
  panels = list(),
  ...,
  theme = c("light-spaced", "light", "abyss", "abyss-spaced", "dark", "vs", "dracula",
    "replit"),
  add_tab = new_add_tab_plugin(),
  width = NULL,
  height = NULL,
  elementId = NULL
)
```

Arguments

panels	An unnamed list of `panel()`.
...	Other options. See https://dockview.dev/docs/api/dockview/options/.
theme	Theme. One of `c("abyss", "dark", "light", "vs", "dracula", "replit")`.

add_tab	Globally controls the add tab behavior. List with enable and callback. Enable is a boolean, default to FALSE and callback is a JavaScript function passed with JS. See default_add_tab_callback(). By default, the callback sets a Shiny input input[["<dock_ID>_panel-to-add"]] so you can create observers with custom logic.
width	Widget width.
height	Widget height.
elementId	When used outside Shiny.

Value

An HTML widget object.

Examples in Shinylive

example-1 Open in Shinylive

---

dock_view_plugins *Create dock view plugins*

Description

Create plugins to enable additional functionality in dock view interfaces. Currently supports "add_tab" and "remove_tab" plugins.

Usage

```
new_dock_view_plugin(type, ...)

## S3 method for class 'add_tab'
new_dock_view_plugin(type, enable = FALSE, callback = NULL, ...)

## S3 method for class 'remove_tab'
new_dock_view_plugin(type, enable = FALSE, callback = NULL, mode = "auto", ...)

new_add_tab_plugin(enable = FALSE, callback = NULL, ...)

new_remove_tab_plugin(enable = FALSE, callback = NULL, mode = "auto", ...)
```

Arguments

type	Character string specifying the plugin type.
...	Additional plugin configuration arguments.
enable	Logical, whether the plugin functionality is enabled.
callback	Optional JavaScript function. If NULL and enable = TRUE, a default callback is used.
mode	For remove_tab plugins only. One of "auto" or "manual".

Value

A dock view plugin object of class `add_tab` or `remove_tab`, depending on the choosen 'type".

Examples

```
# Add tab plugin
new_dock_view_plugin("add_tab", enable = TRUE)
new_add_tab_plugin(enable = TRUE)  # convenience function

# Remove tab plugin
new_dock_view_plugin("remove_tab", enable = TRUE, mode = "auto")
new_remove_tab_plugin(enable = TRUE, mode = "manual")  # convenience function
```

`dock_view_proxy` *Create a proxy object to modify an existing dockview instance*

Description

This function creates a proxy object that can be used to update an existing dockview instance after it has been rendered in the UI. The proxy allows for server-side modifications of the graph without completely re-rendering it.

Usage

```
dock_view_proxy(id, data = NULL, session = getDefaultReactiveDomain())
```

Arguments

`id`	Character string matching the ID of the dockview instance to be modified.
`data`	Unused parameter (for future compatibility).
`session`	The Shiny session object within which the graph exists. By default, this uses the current reactive domain.

Value

A proxy object of class "dock_view_proxy" that can be used with dockview proxy methods such as `add_panel()`, `remove_panel()`, etc. It contains:

- `id`: The ID of the dockview instance.
- `session`: The Shiny session object.

dockViewOutput *Shiny bindings for dock_view*

Description

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

Usage

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

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

renderDockView(expr, env = parent.frame(), quoted = FALSE)

render_dock_view(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 dock_view
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

dockViewOutput and dock_view_output return a Shiny output function that can be used in the UI definition. renderDockView and render_dock_view return a Shiny render function that can be used in the server definition to render a dock_view element.

get_dock *get dock*

Description

get dock

get dock panels

get dock panels ids

get dock active group

get dock grid

get dock groups

get dock groups ids

get dock groups panels

get active views

get active panel

save a dock

restore a dock

Usage

```
get_dock(dock)

get_panels(dock)

get_panels_ids(dock)

get_active_group(dock)

get_grid(dock)

get_groups(dock)

get_groups_ids(dock)

get_groups_panels(dock)

get_active_views(dock)

get_active_panel(dock)

save_dock(dock)

restore_dock(dock, data)
```

Arguments

dock	Dock proxy created with dock_view_proxy().
data	Data representing a serialised dock object.

Value

get_dock returns a list of 3 elements:

- grid: a list representing the dock layout.
- panels: a list having the same structure as panel() composing the dock.
- activeGroup: the current active group (a string).

Each other function allows to deep dive into the returned value of get_dock(). get_panels() returns the panels element of get_dock(). get_panels_ids() returns a character vector containing all panel ids from get_panels(). get_active_group() extracts the activeGroup component of get_dock() as a string. get_active_views() is a convenience function that returns the active view in each group. get_active_panel() is a convenience function that returns the active panel in the active group. get_grid() returns the grid element of get_dock() which is a list. get_groups() returns a list of panel groups from get_grid(). get_groups_ids() returns a character vector of groups ids from get_groups(). get_groups_panels() returns a list of character vector containing the ids of each panel within each group. save_dock() and restore_dock() are used for their side effect to allow to respectively serialise and restore a dock object.

Note

Only works with server side functions like add_panel. Don't call it from the UI.

---

is_add_tab_plugin *Check if object is an add tab plugin*

Description

Check if object is an add tab plugin

Usage

```
is_add_tab_plugin(x)
```

Arguments

x	An object to test.

Value

Logical value.

is_dock_view_plugin *Check if object is a dock view plugin of specific type*

Description

Check if object is a dock view plugin of specific type

Usage

```
is_dock_view_plugin(x, type)
```

Arguments

x	An object to test.
type	Character string specifying plugin type ("add_tab" or "remove_tab").

Value

Logical value indicating whether the object is the specified plugin type.

is_remove_tab_plugin *Check if object is a remove tab plugin*

Description

Check if object is a remove tab plugin

Usage

```
is_remove_tab_plugin(x)
```

Arguments

x	An object to test.

Value

Logical value.

panel *Dock panel*

Description

Create a panel for use within a dock_view() widget. Panels are the main container components that can be docked, dragged, resized, and arranged within the dockview interface.

Usage

```
panel(
  id,
  title,
  content,
  active = TRUE,
  remove = new_remove_tab_plugin(),
 style = list(padding = "10px", overflow = "auto", height = "100%", margin = "10px"),
  ...
)
```

Arguments

id	Panel unique id.
title	Panel title.
content	Panel content. Can be a list of Shiny tags.
active	Is active?
remove	List with two fields: enable and mode. Enable is a boolean and mode is one of manual, auto (default to auto). In auto mode, dockview JS removes the panel when it is closed and all its content. If you need more control over the panel removal, set it to manual so you can explicitly call remove_panel() and perform other tasks. On the server side, a shiny input is available input[["<dock_ID>_panel-to-remove"]] so you can create observers with custom logic.
style	List of CSS style attributes to apply to the panel content. See defaults
...	Other options passed to the API. See https://dockview.dev/docs/api/dockview/panelApi/. If you pass position, it must be a list with 2 fields: • referencePanel: reference panel id. • direction: one of above, below, left, right or within (above, below, left, right put the panel in a new group, while within puts the panel after its reference panel in the same group). Position is relative to the reference panel target.

Value

A list representing a panel object to be consumed by dock_view:

- id: unique panel id (string).
- title: panel title (string).
- content: panel content (`shiny.tag.list` or single `shiny.tag`).
- active: whether the panel is active or not (boolean).
- ...: extra parameters to pass to the API.

`update_dock_view` *Update options for dockview instance*

Description

This does not rerender the widget, just update options like global theme.

Usage

```
update_dock_view(dock, options)
```

Arguments

`dock`	Dock proxy created with `dock_view_proxy()`.
`options`	List of options for the dock_view instance.

Value

This function is called for its side effect. It sends a message to JavaScript through the current websocket connection, leveraging the shiny session object.

Index

add_panel, 2, *9*

default_add_tab_callback, 3
default_add_tab_callback(), *5*
default_remove_tab_callback, 4
dock_view, 4, *12*
dock_view(), *11*
dock_view_output *(*dockViewOutput*)*, 7
dock_view_plugins, 5
dock_view_proxy, 6
dock_view_proxy(), *2*, *8*, *12*
dockViewOutput, 7

get_active_group *(*get_dock*)*, 7
get_active_group(), *9*
get_active_panel *(*get_dock*)*, 7
get_active_panel(), *9*
get_active_views *(*get_dock*)*, 7
get_active_views(), *9*
get_dock, 7
get_dock(), *9*
get_grid *(*get_dock*)*, 7
get_grid(), *9*
get_groups *(*get_dock*)*, 7
get_groups(), *9*
get_groups_ids *(*get_dock*)*, 7
get_groups_ids(), *9*
get_groups_panels *(*get_dock*)*, 7
get_groups_panels(), *9*
get_panels *(*get_dock*)*, 7
get_panels(), *9*
get_panels_ids *(*get_dock*)*, 7
get_panels_ids(), *9*

is_add_tab_plugin, 9
is_dock_view_plugin, 10
is_remove_tab_plugin, 10

JS, *5*

move_group *(*add_panel*)*, 2
move_group2 *(*add_panel*)*, 2
move_panel *(*add_panel*)*, 2

new_add_tab_plugin *(*dock_view_plugins*)*, 5
new_dock_view_plugin *(*dock_view_plugins*)*, 5
new_remove_tab_plugin *(*dock_view_plugins*)*, 5

panel, *2*, 11
panel(), *3*, *4*, *9*

remove_panel *(*add_panel*)*, 2
render_dock_view *(*dockViewOutput*)*, 7
renderDockView *(*dockViewOutput*)*, 7
restore_dock *(*get_dock*)*, 7
restore_dock(), *9*

save_dock *(*get_dock*)*, 7
save_dock(), *9*
select_panel *(*add_panel*)*, 2
set_panel_title *(*add_panel*)*, 2

update_dock_view, 12