Package 'bslibdash'

September 26, 2026

Type Package

Title 'Bootstrap' 5 Dashboard Framework for 'shiny' Apps

Version 0.7.5

Description Provides a dashboard layer for 'shiny' applications built on 'bslib' and 'Bootstrap' 5. Includes a dashboard page shell, sidebar navigation, cards, value boxes, header drop-down menus and feedback components that inherit the active 'bslib' theme and follow 'Bootstrap' design patterns. Function names mirror those of the 'shinydashboard' package wherever the underlying concepts are shared, allowing existing applications to migrate with minimal changes.

License MIT + file LICENSE

URL https://github.com/Novartis/bslibdash, https://opensource.nibr.com/bslibdash/

BugReports https://github.com/Novartis/bslibdash/issues

Depends R (>= 4.1.0)

Imports utils, shiny (>= 1.0.5), shinyjs, htmltools, bslib (>= 0.6.0), bsicons, rlang, glue, sass

Suggests testthat (>= 3.0.0), withr, knitr, lintr, rmarkdown, covr

VignetteBuilder knitr

Encoding UTF-8

Language en-GB

Collate 'useful-items.R' 'cards.R' 'sidebar.R' 'body.R' 'layout.R' 'aliases.R' 'brand_bs_theme.R' 'bslibdash-package.R' 'dependencies.R' 'feedbacks.R' 'grid.R' 'icons.R' 'inputs.R' 'sidebar_bslib.R' 'sidebar_user.R' 'utils.R' 'widgets-boxes.R' 'widgets-menu.R' 'widgets-sidebar.R'

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

Config/testthat/edition 3

NeedsCompilation no

Author Alexandros Kouretsis [aut, cre],
Ardalan Mirshani [aut],
Dominik Rafacz [aut],
Novartis Open Source Initiative [cph]

Maintainer Alexandros Kouretsis <alexandros@appsilon.com>

Repository CRAN

Date/Publication 2026-09-26 16:40:19 UTC

Contents

accordion *Accordion container*

Description

Accordion container

Accordion item

Usage

```
accordion(..., id, width = 12, .list = NULL)

accordionItem(..., title, status = NULL)
```

Arguments

`...`	Item content.
`id`	Unique accordion id.
`width`	The width of the accordion.
`.list`	Optional list of accordion items.
`title`	Item title.
`status`	Optional Bootstrap status color. When set, the item border and header are tinted with the matching subtle status hue, aligned with `box(status = ...)`.

Value

`accordion()` returns a `bslib::accordion()` tag that may be included in a Shiny UI.

`accordionItem()` returns a `bslib::accordion_panel()` tag that may be passed to `accordion()`.

Examples

```
accordion(
  id = "filters",
  accordionItem(title = "Date range", shiny::p("Last 30 days")),
  accordionItem(title = "Region", shiny::p("All regions"))
)

accordionItem(
  title = "Advanced settings",
  status = "info",
  shiny::p("Optional controls")
)
```

`actionButton` *Action button/link*

Description

Action button/link

Usage

```
actionButton(
  inputId,
  label,
  icon = NULL,
  width = NULL,
  ...,
  status = NULL,
  outline = FALSE,
  size = NULL,
  flat = FALSE
)
```

Arguments

`inputId`	The input slot used to access the value.
`label`	Button label.
`icon`	Optional icon.
`width`	Button width.
`...`	Named attributes applied to the button.
`status`	Bootstrap status color.
`outline`	Whether to display an outline style.
`size`	Button size.
`flat`	Whether to apply a flat style.

Value

Returns a UI element for an action button. The server value received for the input corresponding to `inputId` will be an integer that increments with each click.

Examples

```
actionButton(
  inputId = "refresh",
  label = "Refresh",
  icon = "arrow-clockwise",
  status = "primary"
)
```

badge *Create dashboard badge item*

Description

Create dashboard badge item

Backwards-compatible alias to `badge()`

Usage

```
badge(..., position = c("left", "right"), color, rounded = FALSE)

dashboardBadge(..., position = c("left", "right"), color, rounded = FALSE)
```

Arguments

`...`	Badge content.
`position`	Badge position: `"left"` or `"right"`.
`color`	Bootstrap status color.
`rounded`	Whether the badge is rounded.

Value

An `htmltools::span()` tag that may be included in a Shiny UI (e.g. inside a `menuItem()` label).

Examples

```
badge("NEW", color = "success", position = "right", rounded = TRUE)
```

box *Create a card*

Description

Create a card

Update a card from the server side

Backwards-compatible alias to `updateBox()`

Usage

```
box(
  ...,
  title = NULL,
  footer = NULL,
  status = NULL,
  background = NULL,
  width = 6,
  height = NULL,
  collapsible = FALSE,
  collapsed = FALSE,
  closable = FALSE,
  maximizable = FALSE,
  icon = NULL,
  label = NULL,
  id = NULL
)

updateBox(
  id,
  action = c("remove", "toggle", "update", "restore", "toggleMaximize"),
  options = NULL,
  session = shiny::getDefaultReactiveDomain()
)

updateCard(
  id,
  action = c("remove", "toggle", "update", "restore", "toggleMaximize"),
  options = NULL,
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

`...`	Contents of the card.
`title`	Optional title.
`footer`	Optional footer.
`status`	Bootstrap status color applied to the **card header** background. Accepted values are Bootstrap semantic names: `"primary"`, `"secondary"`, `"success"`, `"info"`, `"warning"`, `"danger"`, `"light"`, `"dark"`. Unlike `valueBox()` and `infoBox()`, legacy `shinydashboard` color names (e.g. `"aqua"`, `"blue"`) are **not** supported here.
`background`	Bootstrap status color applied to the **entire card** background. Accepts the same values as `status`. When set, the full card surface is coloured rather than just the header.
`width`	Width of the card. An integer 1–12 is treated as Bootstrap grid columns and the card is automatically wrapped in `shiny::column()`, consistent with `valueBox()`

	and `infoBox()`. Any other value (e.g. `"300px"`) is applied as an inline CSS `width`. Use `NULL` when placing the card inside an existing column or a `boxLayout()`.
`height`	Card height.
`collapsible`	Whether the card body can collapse.
`collapsed`	Whether the card starts collapsed.
`closable`	Whether the card can be hidden.
`maximizable`	Whether the card can be displayed full-screen.
`icon`	Header icon tag or icon name.
`label`	Header label content.
`id`	Id of the card created with `box()`.
`action`	Action to trigger.
`options`	List of new options for `action = "update"`.
`session`	Shiny session.

Value

`box()` returns a `bslib::card()` tag, wrapped in a `shiny::column()` when `width` is an integer between 1 and 12.

`updateBox()` and `updateCard()` return nothing. These functions are called for their side-effects.

See Also

Other cards: `boxLayout()`

Examples

```
box(
  "Card body",
  title = "Status",
  status = "primary",
  id = "status_box"
)

boxLayout(
  box("A", title = "Card A"),
  box("B", title = "Card B"),
  type = "deck"
)

if (interactive()) {
shiny::shinyApp(
  ui = bslib::page_fluid(
    shiny::actionButton("toggle", "Toggle card"),
    box("Card body", title = "Status", id = "status_box")
  ),
  server = function(input, output, session) {
    shiny::observeEvent(input$toggle, {
```

```
      updateBox("status_box", action = "toggle", session = session)
    })
  }
)
}
```

---

boxLayout *Container for cards*

Description

Container for cards

Usage

```
boxLayout(..., .list = NULL, type = c("group", "deck", "columns"))
```

Arguments

`...`	Slot for `box()`.
`.list`	Optional list of cards.
`type`	Layout type. One of: • `"group"`: Bootstrap `.card-group` — flex row with no gaps, merged borders, and equal-height columns. • `"deck"`: Bootstrap 5 grid cards — responsive row with gutters and equal heights per row (`.row.row-cols-* + .h-100`). Replaces the removed `.card-deck` from Bootstrap 4. • `"columns"`: **Deprecated.** Bootstrap 5 removed `.card-columns`. A `bslib::layout_column_wrap(` fallback is used, but prefer calling `bslib::layout_column_wrap()` directly for new code.

Value

An `htmltools::div()` tag containing the supplied cards.

See Also

Other cards: `box()`

Examples

```
boxLayout(
  box("Revenue", title = "KPI"),
  box("Trend", title = "Chart"),
  type = "deck"
)
```

brand_bs_theme *Default bslibdash brand theme*

Description

Builds a Bootstrap 5 theme via `bslib::bs_theme()` with the bslibdash brand colour, typography and shape variables.

Usage

```
brand_bs_theme()
```

Details

Component CSS is no longer bundled into the theme: each bslibdash component attaches its own theme-aware `bslib::bs_dependency_defer()` dependency, so rendering a component with any `bslib::bs_theme()` produces the matching bslibdash styles. `brand_bs_theme()` exists to set the Bootstrap-level variables used as defaults by bslibdash page constructors.

`dashboardPage()` applies this theme by default. To customise it, override variables or rules via bslib and pass the result through the page `theme` argument:

```
dashboardPage(
  header = dashboardHeader(),
  sidebar = dashboardSidebar(),
  body = dashboardBody(),
  theme = brand_bs_theme() |>
    bslib::bs_add_variables(primary = "#8B0000")
)
```

Value

Returns a `bslib::bs_theme()` object.

Examples

```
theme <- brand_bs_theme()
class(theme)
```

column *Column system*

Description

Column system

Usage

```
column(width, ..., offset = 0)
```

Arguments

width	The grid width of the column.
...	Elements to include within the column.
offset	The number of columns to offset this column.

Value

A shiny::column() tag that may be included in a Shiny UI.

Examples

```
shiny::fluidRow(
  column(8, shiny::p("Main content")),
  column(4, shiny::p("Side panel"))
)
```

dashboardBody *Dashboard body*

Description

Containers for tab content placed inside dashboardBody():

Usage

```
dashboardBody(...)

tabItems(..., id = "sidebarMenu")

tabItem(tabName = NULL, ...)
```

Arguments

...	Items to put in the container.
id	Shared Shiny id for the sidebar menu and body tabset. Use the same value in sidebarMenu(id), tabItems(id), and updateTabItems(inputId). Defaults to "sidebarMenu".
tabName	The name of a tab.

Details

- tabItems() is the parent container.
- tabItem(tabName) is one tab; tabName must match the corresponding menuItem() tabName.

Value

dashboardBody() returns an htmltools::div() tag that may be passed to dashboardPage().

tabItems() returns an htmltools::div() tag containing a hidden shiny::tabsetPanel().

tabItem() returns an htmltools::tagList() that may be passed to tabItems().

Examples

```
dashboardBody(
  tabItems(
    tabItem(tabName = "overview", shiny::h2("Overview")),
    tabItem(tabName = "reports", shiny::h2("Reports"))
  )
)
tabItems(
  tabItem(tabName = "overview", shiny::p("Overview content")),
  tabItem(tabName = "reports", shiny::p("Reports content"))
)
tabItem(
  tabName = "overview",
  shiny::h2("Overview"),
  shiny::p("This is the overview tab.")
)
```

---

dashboardFooter *Dashboard footer*

Description

Dashboard footer

Usage

```
dashboardFooter(left = NULL, right = NULL, fixed = FALSE)
```

Arguments

left	Left-side footer content.
right	Right-side footer content.
fixed	Whether to mark footer as fixed.

Value

A <footer> tag that may be passed to dashboardPage(), or NULL if left and right are both NULL.

Examples

```
dashboardFooter(
  left = "Copyright (c) 2026",
  right = "Contact: team@example.org"
)
```

---

dashboardHeader *Dashboard navbar*

Description

Dashboard navbar

Usage

```
dashboardHeader(
  ...,
  title = NULL,
  rightUi = NULL,
  sidebarIcon = icon("bars"),
  disable = FALSE,
  .list = NULL
)
```

Arguments

...	Header UI elements rendered on the right side. For migration compatibility, a first unnamed scalar string is treated as title when title is NULL.
title	Dashboard title.
rightUi	Additional right-side UI content. This is equivalent to passing content through ..., and is kept for bs4Dash-style compatibility.
sidebarIcon	Icon of the main sidebar toggle.
disable	Whether to disable and omit the header. When TRUE, dashboardPage() renders without a header bar and the body occupies the freed vertical space.
.list	Optional list of right-side header UI elements, merged with ....

Value

A <nav> tag that may be passed to dashboardPage(), or NULL if disable = TRUE.

Examples

```
dashboardHeader(
  title = "Operations",
  rightUi = dropdownMenu(
    type = "notifications",
    notificationItem("Server restarted", status = "info")
  )
)
```

---

dashboardPage *Branded dashboard page*

Description

A bslib sidebar layout that arranges the four standard dashboard slots. The title argument, when set, replaces the title text inside header if dashboardHeader() was called without one.

Usage

```
dashboardPage(
  header,
  sidebar,
  body,
  title = NULL,
  footer = NULL,
  theme = brand_bs_theme()
)
```

Arguments

header	Slot for dashboardHeader().
sidebar	Slot for dashboardSidebar().
body	Slot for dashboardBody().
title	Page/app title shown in the browser tab and dashboard header.
footer	Optional slot for dashboardFooter().
theme	A bslib theme. Defaults to brand_bs_theme().

Value

A bslib::page() tag that may be passed to the ui argument of shiny::shinyApp().

Examples

```
ui <- dashboardPage(
  header = dashboardHeader(title = "bslibdash dashboard"),
  sidebar = dashboardSidebar(
    sidebarMenu(
      id = "sidebarMenu",
      menuItem("Overview", tabName = "overview")
    )
  ),
  body = dashboardBody(
    tabItems(
      tabItem(
        tabName = "overview",
        box("Overview content", title = "Overview")
      )
    )
  )
)
```

`dashboardSidebar` *Create a dashboard main sidebar*

Description

Create a dashboard main sidebar

Dashboard main sidebar menu

Dashboard sidebar menu item

Dashboard sidebar menu sub-item

Dashboard sidebar menu header

Usage

```
dashboardSidebar(..., disable = FALSE, width = NULL, collapsed = FALSE)

sidebarMenu(..., id = NULL, .list = NULL)

menuItem(
  text,
  ...,
  icon = NULL,
  badgeLabel = NULL,
  badgeColor = "success",
  tabName = NULL,
  selected = NULL,
  expandedName = NULL,
  startExpanded = FALSE,
  condition = NULL
```

```
)

menuSubItem(
  text,
  tabName = NULL,
  icon = bslibdash::icon("angle-double-right"),
  selected = NULL
)

sidebarHeader(title)
```

Arguments

...	menuSubItem() children.
disable	Whether to disable and omit the sidebar.
width	Expanded sidebar width. Numeric values are interpreted as pixels; CSS strings such as "18rem" are passed through.
collapsed	Whether the sidebar starts collapsed on desktop.
id	Shared Shiny id for the sidebar menu and body tabset. Use the same value in sidebarMenu(id), tabItems(id), and updateTabItems(inputId). Defaults to "sidebarMenu".
.list	Optional list of items.
text	Item name.
icon	Icon tag or icon name.
badgeLabel	Optional badge label.
badgeColor	Badge color.
tabName	Matching tabItem() name.
selected	Whether the item starts selected.
expandedName	Optional unique name for the item's collapse panel. If omitted, a stable id is generated from the item text and children. Set explicitly for multiple items with the same text and children.
startExpanded	Whether children start expanded.
condition	Optional display condition stored as a data attribute.
title	Header title.

Value

dashboardSidebar() returns an <aside> tag that may be passed to dashboardPage(), or NULL if disable = TRUE.

sidebarMenu() returns an htmltools::div() tag that may be passed to dashboardSidebar().

menuItem() returns a menu item that may be passed to sidebarMenu().

menuSubItem() returns a menu item that may be passed to menuItem().

sidebarHeader() returns an htmltools::div() tag that may be passed to sidebarMenu().

Examples

```
dashboardSidebar(
  sidebarMenu(
    id = "sidebarMenu",
    sidebarHeader("Main"),
    menuItem("Overview", tabName = "overview", icon = icon("house")),
    menuItem(
      "Reports",
      icon = icon("bar-chart"),
      menuSubItem("Daily", tabName = "reports_daily"),
      menuSubItem("Monthly", tabName = "reports_monthly")
    )
  )
)

sidebarMenu(
  id = "sidebarMenu",
  menuItem("Overview", tabName = "overview", icon = icon("house")),
  menuItem("Settings", tabName = "settings", icon = icon("gear"))
)

menuItem(
  "Reports",
  icon = icon("bar-chart"),
  menuSubItem("Daily", tabName = "reports_daily"),
  menuSubItem("Monthly", tabName = "reports_monthly")
)

menuSubItem(
  "Daily report",
  tabName = "reports_daily",
  icon = bslibdash::icon("angle-double-right")
)

sidebarHeader("Administration")
```

---

dropdownMenu *Dropdown menu*

Description

Dropdown menu

Message item

Notification item

Task item

Usage

```
dropdownMenu(
  ...,
  type = c("messages", "notifications", "tasks"),
  badgeStatus = "primary",
  icon = NULL,
  headerText = NULL,
  .list = NULL,
  href = NULL
)

messageItem(
  from,
  message,
  icon = shiny::icon("user"),
  time = NULL,
  href = NULL,
  image = NULL,
  color = "secondary",
  inputId = NULL
)

notificationItem(
  text,
  icon = bslibdash::icon("exclamation-triangle"),
  status = "success",
  href = NULL,
  inputId = NULL
)

taskItem(text, value = 0, color = "info", href = NULL, inputId = NULL)
```

Arguments

...	Menu item tags such as `messageItem()`, `notificationItem()`, or `taskItem()`.
type	Dropdown menu type. One of `"messages"`, `"notifications"`, or `"tasks"`.
badgeStatus	Bootstrap status color used for the badge count.
icon	Optional custom icon tag.
headerText	Optional text shown at the top of the dropdown panel.
.list	Optional list of menu item tags.
href	Optional URL for the "More" footer link.
from	Who the message is from.
message	Text of the message.
time	Optional message timestamp text.
image	Optional user image URL.

color	Bootstrap status color used for item accents.
inputId	Optional id to make the item behave like an action button.
text	Item text.
status	Bootstrap status color used for the icon.
value	Percent completion value.

Value

dropdownMenu() returns an htmltools::div() tag that may be passed to dashboardHeader().

messageItem() returns a dropdown item that may be passed to dropdownMenu().

notificationItem() returns a dropdown item that may be passed to dropdownMenu().

taskItem() returns a dropdown item that may be passed to dropdownMenu().

Examples

```
dropdownMenu(
  type = "notifications",
  badgeStatus = "warning",
  notificationItem("Backup completed", status = "success"),
  notificationItem("New deployment", status = "info")
)

messageItem(
  from = "Ops bot",
  message = "Deployment finished successfully.",
  time = "5 mins ago",
  color = "success"
)

notificationItem(
  text = "3 new alerts",
  status = "danger"
)

taskItem(
  text = "Data refresh",
  value = 75,
  color = "info"
)
```

---

dropdownMenuOutput *Dropdown menu output*

Description

Dropdown menu output

Usage

```
dropdownMenuOutput(outputId)
```

Arguments

outputId	Output variable name.

Value

A shiny::uiOutput() container to be filled by renderDropdownMenu().

Examples

```
dropdownMenuOutput("alerts_menu")
```

---

icon *Create an bslibdash icon*

Description

Create an bslibdash icon

Usage

```
icon(name, class = NULL, style = NULL, size = NULL, color = NULL, css = NULL)
```

Arguments

name	Name of icon.
class	Additional classes.
style	Optional inline style string or named CSS list.
size	Icon size applied to a wrapping span.
color	Icon color applied to a wrapping span.
css	Named list of CSS properties applied to the icon tag.

Value

An HTML tag object that can be included in a Shiny UI.

Examples

```
if (interactive()) {
  icon("user")
  icon("bar-chart", class = "text-primary", size = "1.25rem")
}
```

infoBox *Info box*

Description

Info box

Usage

```
infoBox(
  title,
  value = NULL,
  subtitle = NULL,
  icon = shiny::icon("bar-chart"),
  color = "aqua",
  width = 4,
  href = NULL,
  fill = FALSE
)
```

Arguments

title	Box title.
value	Value text.
subtitle	Optional subtitle text.
icon	An icon tag, created by shiny::icon() or icon().
color	Box color. Supports both legacy shinydashboard color names (e.g. "aqua", "light-blue", "fuchsia") and Bootstrap semantic names ("primary", "success", "warning", etc.). Legacy names are mapped to fixed hex values; Bootstrap names reference CSS theme variables so they follow the active theme. A contrasting foreground color is computed automatically. Note: box() uses a simpler status parameter that only accepts Bootstrap semantic names and applies them as utility classes.
width	The width of the box in Bootstrap grid columns (1-12). Use NULL when placing the box inside an existing column.
href	Optional URL to link to.
fill	Whether to fill the entire box background with color.

Value

An htmltools::div() tag, wrapped in a shiny::column() unless width is NULL.

Examples

```
infoBox(
  title = "CPU",
  value = "42%",
  subtitle = "Current usage",
  icon = icon("cpu"),
  color = "success"
)
```

`infoBoxOutput` *Info box output*

Description

Info box output

Usage

```
infoBoxOutput(outputId, width = 4)
```

Arguments

`outputId`	Output variable name.
`width`	The width of the box in Bootstrap grid columns (`1-12`). Use `NULL` when placing the output inside an existing column.

Value

A `shiny::uiOutput()` container to be filled by `renderInfoBox()`.

Examples

```
infoBoxOutput("system_status")
```

menuItemOutput *Sidebar menu item output*

Description

Sidebar menu item output

Usage

```
menuItemOutput(outputId)
```

Arguments

outputId	Output variable name.

Value

A `shiny::uiOutput()` container to be filled by `renderMenu()`.

Examples

```
menuItemOutput("dynamic_menu_item")
```

renderDropdownMenu *Render a dropdown menu*

Description

Render a dropdown menu

Usage

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

Arguments

expr	An expression that returns a dropdown menu tag.
env	The parent environment for the reactive expression.
quoted	Is `expr` a quoted expression.

Value

A `shiny::renderUI()` function that may be assigned to an `output` slot paired with `dropdownMenuOutput()`.

Examples

```
if (interactive()) {
shiny::shinyApp(
  ui = bslib::page_fluid(dropdownMenuOutput("alerts_menu")),
  server = function(input, output, session) {
    output$alerts_menu <- renderDropdownMenu({
      dropdownMenu(
        type = "notifications",
        notificationItem("Job finished", status = "success")
      )
    })
  }
)
}
```

`renderInfoBox` *Render an info box*

Description

Render an info box

Usage

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

Arguments

`expr`	An expression that returns an info box tag.
`env`	The parent environment for the reactive expression.
`quoted`	Is `expr` a quoted expression.

Value

A `shiny::renderUI()` function that may be assigned to an `output` slot paired with `infoBoxOutput()`.

Examples

```
if (interactive()) {
shiny::shinyApp(
  ui = bslib::page_fluid(infoBoxOutput("system_status")),
  server = function(input, output, session) {
    output$system_status <- renderInfoBox({
      infoBox("CPU", "42%", icon = icon("cpu"), color = "success")
    })
  }
)
}
```

renderMenu *Render a sidebar menu element*

Description

Render a sidebar menu element

Usage

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

Arguments

expr	An expression that returns a `menuItem()` or `sidebarMenu()` tag.
env	The parent environment for the reactive expression.
quoted	Is `expr` a quoted expression.

Value

A `shiny::renderUI()` function that may be assigned to an `output` slot paired with `menuItemOutput()` or `sidebarMenuOutput()`.

Examples

```
if (interactive()) {
shiny::shinyApp(
  ui = bslib::page_fluid(menuItemOutput("dynamic_menu_item")),
  server = function(input, output, session) {
    output$dynamic_menu_item <- renderMenu({
      menuItem("Overview", tabName = "overview", icon = icon("house"))
    })
  }
)
}
```

renderValueBox *Render a value box*

Description

Render a value box

Usage

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

Arguments

expr	An expression that returns a value box tag.
env	The parent environment for the reactive expression.
quoted	Is expr a quoted expression.

Value

A shiny::renderUI() function that may be assigned to an output slot paired with valueBoxOutput().

Examples

```
if (interactive()) {
shiny::shinyApp(
  ui = bslib::page_fluid(valueBoxOutput("tickets")),
  server = function(input, output, session) {
    output$tickets <- renderValueBox({
      valueBox("128", "Open tickets", icon = icon("inbox"), color = "primary")
    })
  }
)
}
```

---

sidebarMenuOutput *Sidebar menu output*

Description

Sidebar menu output

Usage

```
sidebarMenuOutput(outputId)
```

Arguments

outputId	Output variable name.

Value

A shiny::uiOutput() container to be filled by renderMenu().

Examples

```
sidebarMenuOutput("dynamic_sidebar_menu")
```

`sidebarSearchForm` *Create a search form to place in a sidebar*

Description

A search form consists of a text input field and a search button.

Usage

```
sidebarSearchForm(
  textId,
  buttonId,
  label = "Search...",
  icon = shiny::icon("search")
)
```

Arguments

`textId`	Shiny input ID for the text input box.
`buttonId`	Shiny input ID for the search button.
`label`	Text label to display inside the search box.
`icon`	An icon tag, created by `shiny::icon()` or `icon()`.

Value

A `<form>` tag that may be passed to `dashboardSidebar()`. The server values received for the inputs corresponding to `textId` and `buttonId` will be the search string and the button click count.

Examples

```
sidebarSearchForm(
  textId = "sidebar_search",
  buttonId = "sidebar_search_btn",
  label = "Search records..."
)
```

`sidebarUserPanel` *Sidebar user panel*

Description

Displays a user identity block (avatar, name, optional subtitle) at the top of the dashboard sidebar. Mirrors `shinydashboard::sidebarUserPanel()`.

Usage

```
sidebarUserPanel(name, image = NULL, subtitle = NULL)
```

Arguments

`name`	User name. Required.
`image`	Optional avatar. Either a URL string (rendered as `<img>`) or an icon/htmltools tag. If `NULL`, a default `bi-person-circle` icon is used.
`subtitle`	Optional secondary text shown beneath the name.

Details

When the sidebar is collapsed on desktop, the name and subtitle are hidden and only the circular avatar remains, centered.

Value

An `htmltools::div()` tag that may be passed to `dashboardSidebar()`.

Examples

```
dashboardSidebar(
  sidebarUserPanel(
    name = "Jane Doe",
    subtitle = "Administrator"
  ),
  sidebarMenu(
    menuItem("Overview", tabName = "overview")
  )
)
```

tabBox *Tab box*

Description

Tab box

Usage

```
tabBox(
  ...,
  id = NULL,
  selected = NULL,
  title = NULL,
  width = 6,
  height = NULL,
  side = c("left", "right")
)
```

Arguments

...	shiny::tabPanel() elements to include in the tab box.
id	Input id for the tabset.
selected	The tab value selected on initial load.
title	Optional title shown in the tab box header.
width	The width of the box in Bootstrap grid columns (1-12). Use NULL when placing the box inside an existing column.
height	Optional CSS height value passed to bslib::navset_card_tab().
side	Whether to place tabs on the "left" or "right" side of the header.

Value

A bslib::navset_card_tab() tag, wrapped in a shiny::column() unless width is NULL.

Examples

```
# Basic tab box with a title in the card header
tabBox(
  id = "quarterly",
  title = "Quarterly summary",
  shiny::tabPanel("Q1", shiny::p("Q1 content")),
  shiny::tabPanel("Q2", shiny::p("Q2 content"))
)

# Pre-select a tab and fix the card height (content scrolls inside)
tabBox(
```

```
  selected = "Q2",
  height   = "200px",
  shiny::tabPanel("Q1", shiny::p("Q1 content")),
  shiny::tabPanel("Q2", shiny::p("Q2 content"))
)
```

`tabsetPanel` *bslibdash tabsetPanel*

Description

bslibdash tabsetPanel

Usage

```
tabsetPanel(
  ...,
  id = NULL,
  selected = NULL,
  type = c("tabs", "pills", "hidden"),
  .list = NULL
)
```

Arguments

`...`	`tabPanel()` elements to include in the tabset
`id`	If provided, you can use `input$id` in your server logic to determine which of the current tabs is active. The value will correspond to the `value` argument that is passed to `tabPanel()`.
`selected`	The `value` (or, if none was supplied, the `title`) of the tab that should be selected by default. If `NULL`, the first tab will be selected.
`type`	`"tabs"` Standard tab look `"pills"` Selected tabs use the background fill color `"hidden"` Hides the selectable tabs. Use `type = "hidden"` in conjunction with `tabPanelBody()` and `updateTabsetPanel()` to control the active tab via other input controls. (See example below)
`.list`	Optional list of tab panels.

Value

A `shiny::tabsetPanel()` tag that may be included in a Shiny UI.

Examples

```
tabsetPanel(
  id = "tabs",
  shiny::tabPanel("Overview", shiny::p("Overview content")),
  shiny::tabPanel("Details", shiny::p("Detail content"))
)
```

`toast` *Create a toast*

Description

Renders a notification through `shiny::showNotification()`. The default Shiny / bslib notification chrome (background, border, position, status color, dismiss button) is used as-is; `toast()` only adds support for the extra parameters below — header (title + optional icon), body, and subtitle footnote — emitted as flat blocks so they do not produce a card-in-notification (box-in-box) look. The `type` option in `options` controls the status color via Shiny's own notification statuses (`"default"`, `"message"`, `"warning"`, `"error"`).

Usage

```
toast(
  title,
  body = NULL,
  subtitle = NULL,
  options = NULL,
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

`title`	Toast title.
`body`	Body content.
`subtitle`	Toast subtitle. Rendered as an additional muted body block below `body` (footnote).
`options`	Toast options. Supports `delay` (auto-hide duration in milliseconds), `autohide` (set `FALSE` to disable auto-hide), `icon` (optional header icon tag), `close` (show the close button, default `TRUE`), and `type` (Shiny notification status: one of `"default"`, `"message"`, `"warning"`, `"error"`).
`session`	Shiny session object.

Value

The notification ID (string) returned by `shiny::showNotification()`, which can be used with `shiny::removeNotification()`.

Examples

```
if (interactive()) {
shiny::shinyApp(
  ui = bslib::page_fluid(shiny::actionButton("show_toast", "Show toast")),
  server = function(input, output, session) {
    shiny::observeEvent(input$show_toast, {
      toast(
        title = "Heads up",
        body = "Background job finished with a warning.",
        options = list(type = "warning", delay = 3000),
        session = session
      )
    })
  }
)
}
```

`updateTabItems` *Update the selected sidebar tab item*

Description

This updates the hidden body tabset used by `tabItems()` and keeps the sidebar active item synchronized with the selected tab.

Usage

```
updateTabItems(
  session = shiny::getDefaultReactiveDomain(),
  inputId,
  selected = NULL
)
```

Arguments

`session`	Shiny session.
`inputId`	The shared `id` used for `sidebarMenu(id)`, `tabItems(id)`, and `updateTabItems(inputId)`. All three must use the same value.
`selected`	Name of the tab to select.

Value

nothing. This function is called for its side-effects.

Examples

```
if (interactive()) {
shiny::shinyApp(
  ui = dashboardPage(
    header = dashboardHeader(title = "Demo"),
    sidebar = dashboardSidebar(
      sidebarMenu(
        id = "sidebarMenu",
        menuItem("Overview", tabName = "overview"),
        menuItem("Reports", tabName = "reports")
      )
    ),
    body = dashboardBody(
      shiny::actionButton("go_reports", "Go to reports"),
      tabItems(
        tabItem(tabName = "overview", shiny::h2("Overview")),
        tabItem(tabName = "reports", shiny::h2("Reports"))
      )
    )
  ),
  server = function(input, output, session) {
    shiny::observeEvent(input$go_reports, {
      updateTabItems(session, inputId = "sidebarMenu", selected = "reports")
    })
  }
)
}
```

`valueBox` *Value box*

Description

Value box

Usage

```
valueBox(value, subtitle, icon = NULL, color = "aqua", width = 4, href = NULL)
```

Arguments

`value`	The value to display in the box.
`subtitle`	Subtitle text.
`icon`	An icon tag, created by `shiny::icon()` or `icon()`.
`color`	Box color. Supports both legacy `shinydashboard` color names (e.g. `"aqua"`, `"light-blue"`, `"fuchsia"`) and Bootstrap semantic names (`"primary"`, `"success"`, `"warning"`, etc.). Legacy names are mapped to fixed hex values; Bootstrap

	names reference CSS theme variables so they follow the active theme. A contrasting foreground color is computed automatically. Note: `box()` uses a simpler `status` parameter that only accepts Bootstrap semantic names and applies them as utility classes.
`width`	The width of the box in Bootstrap grid columns (1-12). Use `NULL` when placing the box inside an existing column.
`href`	Optional URL to link to.

Value

A `bslib::value_box()` tag, wrapped in a `shiny::column()` unless `width` is `NULL`.

Examples

```
valueBox(
  value = "128",
  subtitle = "Open tickets",
  icon = icon("inbox"),
  color = "primary"
)
```

---

valueBoxOutput *Value box output*

Description

Value box output

Usage

```
valueBoxOutput(outputId, width = 4)
```

Arguments

`outputId`	Output variable name.
`width`	The width of the box in Bootstrap grid columns (1-12). Use `NULL` when placing the output inside an existing column.

Value

A `shiny::uiOutput()` container to be filled by `renderValueBox()`.

Examples

```
valueBoxOutput("tickets")
```

Index