Package ‘bs4Dash’

May 5, 2022

Type  Package
Title  A 'Bootstrap 4' Version of 'shinydashboard'
Version  2.1.0
Maintainer  David Granjon <dgranjon@ymail.com>
Description  Make 'Bootstrap 4' Shiny dashboards. Use the full power of 'AdminLTE3', a dashboard template built on top of 'Bootstrap 4' <https://github.com/ColorlibHQ/AdminLTE>.
BugReports  https://github.com/RinteRface/bs4Dash/issues
License  GPL (>= 2) | file LICENSE
Imports  shiny (>= 1.6.0),
  htmltools (>= 0.5.1.1),
  jsonlite (>= 0.9.16),
  fresh,
  waiter (>= 0.2.3),
  httpuv (>= 1.5.2),
  lifecycle,
  bslib (>= 0.2.4),
  http
Suggests  knitr,
  rmarkdown,
  testthat (>= 2.1.0),
  covr,
  golem,
  DT,
  thematic (>= 0.1.2)
Encoding  UTF-8
RoxygenNote  7.1.2
VignetteBuilder  knitr
Collate 'feedbacks.R'
  'useful-items.R'
  'tabs.R'
  'render-functions.R'
  'cards.R'
  'dashboardSidebar.R'
  'dashboardBody.R'
  'dashboardFooter.R'
  'dashboardControlbar.R'
  'dashboardHeader.R'
  'dashboardPage.R'
  'aliases.R'
  'auto-color.R'
  'bs4Dash-package.r'
  'bs4DashGallery.R'
  'deps.R'
  'grid.R'
  'inputs.R'
  'skinSelector.R'
  'utils.R'

R topics documented:

RdMacros  lifecycle

R topics documented:

  actionButton ................................................. 3
  appButton ................................................... 6
  attachmentBlock ........................................... 7
  bs4Accordian ............................................... 9
  bs4Badge .................................................... 12
  bs4Callout .................................................. 13
  bs4CardLabel ............................................... 15
  bs4CardLayout ............................................ 15
  bs4CardSidebar ........................................... 17
  bs4Carousel ................................................ 20
  bs4DashBody ............................................... 21
  bs4DashBrand ............................................... 22
  bs4DashControlbar ....................................... 23
  bs4DashFooter ............................................. 25
  bs4DashGallery ........................................... 26
  bs4DashNavbar ............................................. 26
  bs4DashPage ............................................... 28
  bs4DashSidebar ........................................... 31
  bs4DropdownMenu .......................................... 40
  bs4Jumbotron .............................................. 45
  bs4ListGroup ............................................... 46
  bs4Loading ................................................ 49
  bs4ProgressBar .......................................... 50
**ActionButton**

**Bootstrap 4 Action button/link**

**Description**

Creates an action button or link whose value is initially zero, and increments by one each time it is pressed.
Usage

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

Arguments

- **inputId**  
The `input` slot that will be used to access the value.
- **label**  
The contents of the button or link—usually a text label, but you could also use any other HTML, like an image.
- **icon**  
An optional [icon()] to appear on the button.
- **width**  
The width of the input, e.g. `"400px"`, or `"100` see [validateCssUnit()].
- **...**  
Named attributes to be applied to the button or link.
- **status**  
Button status color. Valid statuses are defined as follows:
  - primary: `#007bff`.
  - secondary: `#6c757d`.
  - info: `#17a2b8`.
  - success: `#28a745`.
  - warning: `#ffc107`.
  - danger: `#dc3545`.
- **gradient**  
Whether to apply gradient to color. Default to FALSE.
- **outline**  
Whether to display an outline style. Status must not be NULL if TRUE. Default to FALSE.
- **size**  
Button size. Default to NULL. Possible choices: c("lg", "sm", "xs").
- **flat**  
Whether to apply a flat style. Default to FALSE.

Server value

An integer of class `"shinyActionButtonValue"`. This class differs from ordinary integers in that a value of 0 is considered "falsy". This implies two things: * Event handlers (e.g., [observeEvent()], [eventReactive()]) won’t execute on initial load. * Input validation (e.g., [req()], [need()]) will fail on initial load.

Note

One may also pass the status directly via the ... parameter using `class = "btn-primary"`, for the primary status for instance. Same thing for other styles like the size.
See Also

[observeEvent()] and [eventReactive()]

Examples

```r
## Only run examples in interactive R sessions
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(
        title = bs4DashBrand(
          title = "My dashboard",
          color = "primary",
          src = "https://adminlte.io/themes/v3",
          image = "https://adminlte.io/themes/v3/dist/img/AdminLTELogo.png"
        )
      ),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        sliderInput("obs", "Number of observations", 0, 1000, 500),
        actionButton("goButton", "Go!",
          status = "danger",
          outline = TRUE,
          flat = TRUE,
          size = "lg"
        ),
        plotOutput("distPlot")
      ),
      controlbar = dashboardControlbar(),
      title = "DashboardPage"
    ),
    server = function(input, output) {
      output$distPlot <- renderPlot({
        # Take a dependency on input$goButton. This will run once initially,
        # because the value changes from NULL to 0.
        input$goButton
        # Use isolate() to avoid dependency on input$obs
        dist <- isolate(rnorm(input$obs))
        hist(dist)
      })
    }
  }
}
```

## Example of adding extra class values
`actionButton("largeButton", "Large Primary Button", class = "btn-primary btn-lg")`
**appButton**

*AdminLTE2 special large button*

**Description**

Create a large button ideal for web applications but identical to the classic Shiny action button.

**Usage**

```r
appButton(..., inputId, label, icon = NULL, width = NULL, color = NULL)
```

**Arguments**

- `...` Named attributes to be applied to the button or link.
- `inputId` The input slot that will be used to access the value.
- `label` The contents of the button or link–usually a text label, but you could also use any other HTML, like an image.
- `icon` An optional `icon()` to appear on the button.
- `width` The width of the input, e.g. '400px', or '100%'; see `validateCssUnit()`.
- `color` Button background color. Valid statuses are defined as follows:
  - primary: #007bff.
  - secondary: #6c757d.
  - info: #17a2b8.
  - success: #28a745.
  - warning: #ffc107.
  - danger: #dc3545.
  - gray-dark: #343a40.
  - gray: #adb5bd.
  - white: #fff.
  - indigo: #6610f2.
  - lightblue: #3c8dbc.
  - navy: #001f3f.
  - purple: #605ca8.
  - fuchsia: #f012be.
  - pink: #e83e8c.
  - maroon: #d81b60.
  - orange: #ff851b.
  - lime: #01ff70.
  - teal: #39cccc.
  - olive: #3d9970.
attachmentBlock

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "App Buttons",
          status = NULL,
          appButton(
            inputId = "myAppButton",
            label = "Users",
            icon = icon("users"),
            color = "orange",
            dashboardBadge(textOutput("btnVal"), color = "primary")
          )
        ),
        title = "App buttons"
      ),
      server = function(input, output) {
        output$btnVal <- renderText(input$myAppButton)
      }
    )
  )
}
```

attachmentBlock

AdminLTE3 attachment container

Description

attachmentBlock create an attachment container, nice to wrap articles... and insert in a box.

Usage

attachmentBlock(..., image, title = NULL, href = NULL)
attachmentBlock

Arguments

... Any element.
image url or path to the image.
title Attachment title.
href External link.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

Other boxWidgets: bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardDropdown(), cardProfile(), descriptionBlock(), userPost()

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "attachmentBlock example",
          attachmentBlock(
            image = "https://adminlte.io/themes/v3/dist/img/user1-128x128.jpg",
            title = "Test",
            href = "https://google.com",
            "This is the content"
          )
        )
      ),
      title = "attachmentBlock"
    ),
    server = function(input, output) {
    }
  )
}
```
bs4Accordion

Bootstrap 4 accordion container

Description

- accordion creates an accordion container. Accordions are part of collapsible elements.
- accordionItem is to be inserted in a accordion.
- updateAccordion toggles an accordion on the client.

Usage

bs4Accordion(..., id, width = 12)

bs4AccordionItem(
    ...,
    title,
    status = NULL,
    collapsed = TRUE,
    solidHeader = TRUE
)

updateAccordion(id, selected, session = shiny::getDefaultReactiveDomain())

accordion(..., id, width = 12)

accordionItem(..., title, status = NULL, collapsed = TRUE, solidHeader = TRUE)

Arguments

- ... slot for accordionItem.
- id Accordion to target.
- width The width of the accordion.
- title Optional title.
- status The status of the item. This determines the item’s background color. Valid statuses are defined as follows:
  - primary: #007bff.
  - secondary: #6c757d.
  - info: #17a2b8.
  - success: #28a745.
  - warning: #ffc107.
  - danger: #dc3545.
  - gray-dark: #343a40.
  - gray: #adb5bd.
• white: #fff.
• indigo: #6610f2.
• lightblue: #3c8dbc.
• navy: #001f3f.
• purple: #605ca8.
• fuchsia: #f012be.
• pink: #e83e8c.
• maroon: #d81b60.
• orange: #ff851b.
• lime: #01ff70.
• teal: #39ccc.
• olive: #3d9970.

collapsed  If TRUE, start collapsed. This must be used with collapsible=TRUE.
solidHeader Should the header be shown with a solid color background?
selected Index of the newly selected accordionItem.
session Shiny session object.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if (interactive()) {
library(shiny)
library(bs4Dash)

shinyApp(
ui = dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
        accordion(
            id = "accordion1",
            accordionItem(
                title = "Accordion 1 Item 1",
                status = "danger",
                collapsed = TRUE,
                "This is some text!"
            ),
            accordionItem(
                title = "Accordion 1 Item 2",
                status = "indigo",
                collapsed = FALSE,
                "This is some text!"
            )
        )
    )
)
id = "accordion2",
  accordionItem(
    title = "Accordion 2 Item 1",
    status = "info",
    collapsed = TRUE,
    "This is some text!"
  ),
  accordionItem(
    title = "Accordion 2 Item 2",
    status = "success",
    collapsed = FALSE,
    "This is some text!"
  )
),
},
title = "Accordion"
),
server = function(input, output) { }
}

# Update accordion
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        radioButtons("controller", "Controller", choices = c(1, 2)),
        br(),
        accordion(
          id = "accordion1",
          accordionItem(
            title = "Accordion 1 Item 1",
            status = "danger",
            collapsed = TRUE,
            "This is some text!"
          ),
          accordionItem(
            title = "Accordion 1 Item 2",
            status = "warning",
            collapsed = TRUE,
            "This is some text!"
          )
        ),
        title = "Update Accordion"
      ),
      server = function(input, output, session) {

observeEvent(input$controller, {
    updateAccordion(id = "accordion1", selected = input$controller)
})
observe(print(input$accordion1))
observeEvent(input$accordion1, {
    showNotification(sprintf("You selected accordion N° %s", input$accordion1), type = "message")
})
}
}

bs4Badge

Create a Bootstrap 4 dashboard badge item

Description

dashboardBadge creates a badge. It may be inserted in any element like inside a actionButton or a dashboardSidebar.

Usage

bs4Badge(..., color, position = c("left", "right"), rounded = FALSE)

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

Arguments

... Badge content.

color Badge color. Valid colors:

- primary: #007bff.
- secondary: #6c757d.
- info: #17a2b8.
- success: #28a745.
- warning: #ffc107.
- danger: #dc3545.

position Badge position: "left" or "right".

rounded Whether the badge is rounded instead of square. FALSE by default.

Author(s)

David Granjon, <dgranjon@ymail.com>
Examples

```r
if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        dashboardBadge("Badge 1", color = "danger"),
        actionButton(
          inputId = "badge",
          label = "Hello",
          icon = NULL,
          width = NULL,
          dashboardBadge(1, color = "primary")
        )
      ),
      server = function(input, output) { }
    )
  )
}
```

bs4Callout Create a Bootstrap 4 callout

Description

AdminLTE3 callout

Usage

```r
bs4Callout(
  ..., title, status = c("warning", "danger", "info", "success"),
  width = 6, elevation = NULL
)
```

callout(
  ..., title, status = c("warning", "danger", "info", "success"),
  width = 6, elevation = NULL
)```
Arguments

... Callout content.

title Callout title.

status Callout status. Valid statuses:
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffc107.
  • danger: #dc3545.

width Callout width. Between 1 and 12.

elevation Callout elevation.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Callout",
      body = bs4DashBody(
        title = "Callouts",
        callout(
          title = "I am a danger callout!",
          elevation = 4,
          status = "danger",
          "There is a problem that we need to fix.
          A wonderful serenity has taken possession of
          my entire soul, like these sweet mornings of
          spring which I enjoy with my whole heart."
        ),
        callout(
          title = "I am a danger callout!",
          status = "warning",
          "This is a yellow callout."
        )
      )
    ),
    server = function(input, output) {}  
  )
}
bs4CardLabel

Create a label for Bootstrap 4 card

Description

Create a label for Bootstrap 4 card
Alias to bs4CardLabel See bs4CardLabel for full details
Alias to bs4CardLabel See bs4CardLabel for full details

Usage

bs4CardLabel(text, status, tooltip = NULL)
cardLabel(text, status, tooltip = NULL)
boxLabel(text, status, tooltip = NULL)

Arguments

text     Label text. In practice only few letters or a number.
status   label color status. See getAdminLTEColors.
tooltip  Label tooltip text on hover.

See Also

Other boxWidgets: attachmentBlock(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardDropdown(), cardProfile(), descriptionBlock(), userPost()

bs4CardLayout

Bootstrap 4 container for cards

Description

Bootstrap 4 container for cards
Alias to bs4CardLayout See bs4CardLayout for full details

Usage

bs4CardLayout(..., type = c("group", "deck", "columns"))
boxLayout(..., type = c("group", "deck", "columns"))
bs4CardLayout

Arguments

... Slot for bs4Dash cards.

type Container type. See https://getbootstrap.com/docs/4.0/components/card/#card-layout for more details.

Note

Cards must have width argument set to NULL.

See Also

Other cards: bs4SocialCard(), bs4TabCard(), bs4UserCard(), descriptionBlock(), renderbs4InfoBox(), renderbs4ValueBox()

Examples

if (interactive()) {
  library(shiny)
  library(bs4Dash)

  # with group
  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        boxLayout(
          type = "group",
          lapply(1:4, function(i) {
            box(
              width = NULL,
              title = paste("Card", i),
              closable = FALSE,
              collapsible = FALSE,
              "Lorem ipsum is so fun!"
            )
          })
        ),
        controlbar = dashboardControlbar(),
        title = "Box layout group"
      ),
      server = function(input, output) {}
    )
  )

  # with deck
  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
      })
  )
}
bs4CardSidebar

Create a sidebar for Bootstrap 4 card
bs4CardSidebar

Description
To insert in the sidebar slot of box.

Usage
bs4CardSidebar(
    ..., id = NULL, width = 50, background = "#333a40", startOpen = FALSE, icon = shiny::icon("cogs"), easyClose = TRUE
)

updatebs4CardSidebar(id, session = shiny::getDefaultReactiveDomain())

cardSidebar(
    ..., id = NULL, width = 50, background = "#333a40", startOpen = FALSE, icon = shiny::icon("cogs"), easyClose = TRUE
)

boxSidebar(
    ..., id = NULL, width = 50, background = "#333a40", startOpen = FALSE, icon = shiny::icon("cogs"), easyClose = TRUE
)

updateCardSidebar(id, session = shiny::getDefaultReactiveDomain())

updateBoxSidebar(id, session = shiny::getDefaultReactiveDomain())

Arguments
... Sidebar content.
id Card sidebar id.
width Sidebar opening width in percentage. 50% by default, means the card sidebar will take 50 A numeric value between 25 and 100.
background Sidebar background color. Dark by default.
bs4CardSidebar

- **startOpen** Whether the sidebar is open at start. FALSE by default.
- **icon** Sidebar icon. Expect icon.
- **easyClose** Whether to close sidebar on click outside. Default to TRUE.
- **session** Shiny session object.

**See Also**

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardDropdown(), cardProfile(), descriptionBlock(), userPost()

**Examples**

```r
# Toggle a box sidebar
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      body = dashboardBody(
        box(
          title = "Update box sidebar",
          closable = TRUE,
          width = 12,
          height = "500px",
          solidHeader = FALSE,
          collapsible = TRUE,
          actionButton("update", "Toggle card sidebar"),
          sidebar = boxSidebar(
            id = "mycardsidebar",
            p("Sidebar Content")
          )
        )
      ),
      sidebar = dashboardSidebar()
    ),
    server = function(input, output, session) {
      observe(print(input$mycardsidebar))

      observeEvent(input$update, {
        updateBoxSidebar("mycardsidebar")
      })
    }
  )
}
```
bs4Carousel

**Description**

carousel creates a carousel container to display media content.
carouselItem creates a carousel item to insert in a carousel

**Usage**

bs4Carousel(..., id, indicators = TRUE, width = 12, .list = NULL)

bs4CarouselItem(..., caption = NULL, active = FALSE)

carousel(..., id, indicators = TRUE, width = 12, .list = NULL)

carouselItem(..., caption = NULL, active = FALSE)

**Arguments**

... Element such as images, iframe, ...
id Unique carousel id.
indicators Whether to display left and right indicators.
width Carousel width. Between 1 and 12.
.list Should you need to pass carouselItem via lapply or similar, put these item here instead of passing them in ...
caption Item caption.
active Whether the item is active or not at start.

**Author(s)**

David Granjon, <dgranjon@ymail.com>

**See Also**

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4SocialCard(), bs4Timeline(), cardDropdown(), cardProfile(), descriptionBlock(), userPost()

**Examples**

```r
if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
```
header = dashboardHeader(),
sidebar = dashboardSidebar(),
body = dashboardBody(
carousel(
    id = "mycarousel",
carouselItem(
    caption = "Item 1",
tags$img(src = "https://via.placeholder.com/500")
),
carouselItem(
    caption = "Item 2",
tags$img(src = "https://via.placeholder.com/500")
)
),
title = "Carousel"
),
server = function(input, output) { }
)

bs4DashBody  Bootstrap 4 dashboard body

Description

dashboardBody creates the main body container for a dashboardPage.
tabItems creates a wrapper for multiple tabItem.
tabItem creates a body tab content.

Usage

bs4DashBody(...)

bs4TabItems(...)

bs4TabItem(tabName = NULL, ...)

dashboardBody(...)

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

Arguments

... Contents of the tab.
tabName The name of a tab. This must correspond to the tabName of a sidebar menuItem.
bs4DashBrand

Author(s)

David Granjon. <dgranjon@ymail.com>

See Also

dashboardSidebar

---

bs4DashBrand  
Alternative to simple text title

Description

Alternative to simple text title
Alias to bs4DashBrand See bs4DashBrand for full details

Usage

bs4DashBrand(title, color = NULL, href = NULL, image = NULL, opacity = 0.8)

dashboardBrand(title, color = NULL, href = NULL, image = NULL, opacity = 0.8)

Arguments

title  
Brand title.

color  
Brand color. Valid colors are defined as follows:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffc107.
  • danger: #dc3545.
  • gray-dark: #343a40.
  • gray: #adb5bd.
  • white: #fff.
  • indigo: #6610f2.
  • lightblue: #3c8dbc.
  • navy: #001f3f.
  • purple: #605ca8.
  • fuchsia: #f012be.
  • pink: #e83e8c.
  • maroon: #d81b60.
  • orange: #ff851b.
  • lime: #01ff70.
bs4DashControlbar

- teal: #39cccc.
- olive: #3d9970.

href External link to point to.
image External image location.
opacity Brand opacity: value between 0 and 1.

Value
A title tag to be inserted in the title slot of bs4DashNavbar.

bs4DashControlbar Create a Bootstrap 4 dashboard right sidebar

Description
Build an adminLTE3 dashboard right sidebar

Usage
bs4DashControlbar(
  ..., 
  id = NULL,
  disable = FALSE,
  width = 250,
  collapsed = TRUE,
  overlay = TRUE,
  skin = "dark",
  pinned = NULL
)

colorbarMenu(
  ..., 
  id = NULL,
  selected = NULL,
  type = c("tabs", "pills", "hidden"),
  vertical = FALSE,
  side = "left",
  .list = NULL
)

colorbarItem(title, ..., value = title, icon = NULL)

updateColorbarMenu(
  session = shiny::getDefaultReactiveDomain(),
  inputId,
  selected = NULL
)
updateControlbar(id, session = shiny::getDefaultReactiveDomain())

dashboardControlbar(
  ..., 
  id = NULL, 
  disable = FALSE, 
  width = 250, 
  collapsed = TRUE, 
  overlay = TRUE, 
  skin = "dark", 
  pinned = NULL 
)

Arguments

... Any UI element.

id Controlbar id.

disable If TRUE, the sidebar will be disabled.

width Controlbar width. This must either be a number which specifies the width in pixels, or a string that specifies the width in CSS units. 250 px by default.

collapsed Whether the control bar on the right side is collapsed or not at start. TRUE by default.

overlay Whether the sidebar covers the content when expanded. Default to TRUE.

skin Controlbar skin. "dark" or "light".

pinned Whether to block the controlbar state (TRUE or FALSE). Default to NULL.

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

vertical Whether to displays tabs vertically. Default to FALSE.

side Tabs side: "left" or "right".

.title In case of programmatically generated items. See example.

title Display title for tab

value The value that should be sent when tabsetPanel reports that this tab is selected. If omitted and tabsetPanel has an id, then the title will be used.

icon Optional icon to appear on the tab. This attribute is only valid when using a tabPanel within a navbarPage().

session Shiny session object.

inputId The id of the tabsetPanel, navlistPanel, or navbarPage object.

Author(s)

David Granjon, <dgranjon@ymail.com>
Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        actionButton(inputId = "controlbarToggle", label = "Toggle Controlbar")
      ),
      controlbar = dashboardControlbar(
        id = "controlbar",
        collapsed = FALSE,
        overlay = TRUE
      ),
      title = "updateControlbar"
    ),
    server = function(input, output, session) {
      observeEvent(input$controlbar, {
        if (input$controlbar) {
          showModal(modalDialog(
            title = "Alert",
            "The controlbar is opened."
          )
        )
      })
      }
      }
      observeEvent(input$controlbarToggle, {
        updateControlbar(id = "controlbar", session = session)
      })
      observe({
        print(input$controlbar)
      })
    }
  )
}
```

bs4DashFooter  **Dashboard Footer**

Description

This creates a dashboard footer to insert in `dashboardPage`. 
bs4DashNavbar

Usage

bs4DashFooter(left = NULL, right = NULL, fixed = FALSE)
dashboardFooter(left = NULL, right = NULL, fixed = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>left</td>
<td>Left text.</td>
</tr>
<tr>
<td>right</td>
<td>Right text.</td>
</tr>
<tr>
<td>fixed</td>
<td>Whether to fix the footer. Default to FALSE.</td>
</tr>
</tbody>
</table>

Author(s)

David Granjon, <dgranjon@ymail.com>

bs4DashGallery

Launch the bs4Dash Gallery

Description

A gallery of all components available in bs4Dash.

Usage

bs4DashGallery()

Examples

if (interactive()) {
  bs4DashGallery()
}

bs4DashNavbar

Bootstrap 4 dashboard navbar

Description

dashboardHeader creates an adminLTE3 dashboard navbar to be included in dashboardPage.
bs4DashNavbar

Usage

bs4DashNavbar(
  ..., 
  title = NULL,
  titleWidth = NULL,
  disable = FALSE,
  .list = NULL,
  leftUi = NULL,
  rightUi = NULL,
  skin = "light",
  status = "white",
  border = TRUE,
  compact = FALSE,
  sidebarIcon = shiny::icon("bars"),
  controlbarIcon = shiny::icon("th"),
  fixed = FALSE
)

dashboardHeader(
  ..., 
  title = NULL,
  titleWidth = NULL,
  disable = FALSE,
  .list = NULL,
  leftUi = NULL,
  rightUi = NULL,
  skin = "light",
  status = "white",
  border = TRUE,
  compact = FALSE,
  sidebarIcon = shiny::icon("bars"),
  controlbarIcon = shiny::icon("th"),
  fixed = FALSE
)

Arguments

... Any UI element between left and right Ui.

title Dashboard title (displayed top-left side). Alternatively, use dashboardBrand for more evolved title.

titleWidth This argument is deprecated; bs4Dash (AdminLTE3) title width is tightly related to the sidebar width, contrary to shinydashboard (AdminLTE2).

disable If TRUE, don’t display the header bar.

.list An optional list containing items to put in the header. Same as the ... arguments, but in list format. This can be useful when working with programmatically generated items.

leftUi Custom left Ui content. Any element like dropdownMenu.
rightUi
Custom right Ui content. Any element like dropdownMenu.
skin
Navbar skin. "dark" or "light".
status
Navbar status. Valid statuses are defined as follows:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffc107.
  • danger: #dc3545.
  • gray-dark: #343a40.
  • gray: #adb5bd.
  • white: #fff.
  • indigo: #6610f2.
  • lightblue: #3c8dbc.
  • navy: #001f3f.
  • purple: #605ca8.
  • fuchsia: #f012be.
  • pink: #e83e8c.
  • maroon: #d81b60.
  • orange: #ff851b.
  • lime: #01ff70.
  • teal: #39cccc.
  • olive: #3d9970.
border
Whether to separate the navbar and body by a border. TRUE by default.
compact
Whether items should be compacted. FALSE by default.
sidebarIcon
Icon of the main sidebar toggle.
controlbarIcon
Icon to toggle the controlbar (left).
fixed
Whether to fix the navbar to the top. FALSE by default.

Author(s)
David Granjon, <dgranjon@ymail.com>

bs4DashPage
Create a Bootstrap 4 dashboard page

Description
Build an adminLTE3 dashboard page
Usage

```r
bs4DashPage(
  header,
  sidebar,
  body,
  controlbar = NULL,
  footer = NULL,
  title = NULL,
  skin = NULL,
  freshTheme = NULL,
  preloader = NULL,
  options = NULL,
  fullscreen = FALSE,
  help = FALSE,
  dark = FALSE,
  scrollToTop = FALSE
)
```

dashboardPage(
  header,
  sidebar,
  body,
  controlbar = NULL,
  footer = NULL,
  title = NULL,
  skin = NULL,
  freshTheme = NULL,
  preloader = NULL,
  options = NULL,
  fullscreen = FALSE,
  help = FALSE,
  dark = FALSE,
  scrollToTop = FALSE
)

Arguments

- **header**: Slot for `bs4DashNavbar`.
- **sidebar**: Slot for `bs4DashSidebar`.
- **body**: Slot for `bs4DashBody`.
- **controlbar**: Slot for `bs4DashControlbar` (right side).
- **footer**: Slot for `bs4DashFooter`.
- **title**: App title.
- **skin**: Deprecated skin parameters. See `skinSelector` for live theming.
- **freshTheme**: A skin powered by the fresh package. Not compatible with skin. See `https://dreamrs.github.io/fresh/articles-vars-shinydashboard.html`.
bs4Dash uses waiter (see https://waiter.john-coene.com/#/). Pass a list like `list(html = spin_1(), color = "#333e48")` to configure `waiterShowOnLoad` (refer to the package help for all styles).

`preloader` Extra option to overwrite the vanilla AdminLTE configuration. See https://adminlte.io/themes/AdminLTE/documentation/index.html#adminlte-options. Expect a list.

`options` Whether to allow fullscreen feature in the navbar. Default to FALSE.

`fullscreen` Whether to enable/disable popovers and tooltips. This allows to seamlessly use `tooltip` and `popover` without having to individually toggle them. Default to FALSE. if TRUE, a help icon is display in the navigation bar.

`help` Whether to display toggle to switch between dark and light mode in the dashboardHeader. Default to FALSE, app starts in light mode, with possibility to switch to dark. If TRUE, the app starts in dark with possibility to switch back to light. If NULL, not toggle is shown and the app starts in light, as it has always been.

`dark` Whether to display a scroll to top button whenever the page height is too large. Default to FALSE.

`scrollToTop` Whether to display a scroll to top button whenever the page height is too large. Default to FALSE.

**Author(s)**

David Granjon, <dgranjon@ymail.com>

**Examples**

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)
  library(fresh)

  shinyApp(
    ui = dashboardPage(
      freshTheme = create_theme(
        bs4dash_vars(
          navbar_light_color = "#bec5cb",
          navbar_light_active_color = "#FFF",
          navbar_light_hover_color = "#FFF"
        ),
        bs4dash_yiq(
          contrasted_threshold = 10,
          text_dark = "#FFF",
          text_light = "#272c30"
        ),
        bs4dash_layout(
          main_bg = "#353c42"
        ),
        bs4dash_sidebar_light(
          bg = "#272c30",
          color = "#bec5cb",
          hover_color = "#FFF",
          submenu_bg = "#272c30",
```
bs4DashSidebar

Create a Bootstrap 4 dashboard main sidebar

Description

dashboardSidebar creates an adminLTE3 dashboard main sidebar to insert in the sidebar slot of dashboardPage.
updateSidebar toggles a dashboardSidebar on the client.
sidebarMenu creates a menu for dashboardSidebar.
menuItem creates an item to put in sidebarMenu.
menuSubItem creates an item to put in menuItem.
sidebarHeader creates a header to put in dashboardSidebar.
sidebarUserPanel creates a user Panel to put in dashboardSidebar.
updateTabItems controls the active tab of tabItems from the server. It behaves just like updateTabsetPanel.
Usage

bs4DashSidebar(
    ..., 
    disable = FALSE, 
    width = NULL, 
    skin = "dark", 
    status = "primary", 
    elevation = 4, 
    collapsed = FALSE, 
    minified = TRUE, 
    expandOnHover = TRUE, 
    fixed = TRUE, 
    id = NULL, 
    customArea = NULL
)

updatebs4Sidebar(id, session = shiny::getDefaultReactiveDomain())

bs4SidebarMenu(
    ..., 
    id = NULL, 
    .list = NULL, 
    flat = FALSE, 
    compact = FALSE, 
    childIndent = TRUE, 
    legacy = FALSE
)

bs4SidebarMenuItem(
    text, 
    ..., 
    icon = NULL, 
    badgeLabel = NULL, 
    badgeColor = "success", 
    tabName = NULL, 
    href = NULL, 
    newTab = TRUE, 
    selected = NULL, 
    expandedName = as.character(gsub("[:space:]", ", text)), 
    startExpanded = FALSE, 
    condition = NULL
)

bs4SidebarMenuSubItem(
    text, 
    tabName = NULL, 
    href = NULL, 
    newTab = NULL,
bs4DashSidebar

icon = shiny::icon("angle-double-right"),
       selected = NULL
)

bs4SidebarHeader(title)

bs4SidebarUserPanel(name, image = NULL)

updatebs4TabItems(
    session = shiny::getDefaultReactiveDomain(),
    inputId,
    selected = NULL
)

dashboardSidebar(
    ...,  
    disable = FALSE,
    width = NULL,
    skin = "dark",
    status = "primary",
    elevation = 4,
    collapsed = FALSE,
    minified = TRUE,
    expandOnHover = TRUE,
    fixed = TRUE,
    id = NULL,
    customArea = NULL
)

updateSidebar(id, session = shiny::getDefaultReactiveDomain())

sidebarHeader(title)

sidebarMenu(
    ...,  
    id = NULL,
    .list = NULL,
    flat = FALSE,
    compact = FALSE,
    childIndent = TRUE,
    legacy = FALSE
)

sidebarUserPanel(name, image = NULL)

menuItem(
    text,
    ...,
icon = NULL,
badgeLabel = NULL,
badgeColor = "success",
tabName = NULL,
href = NULL,
newTab = TRUE,
selected = NULL,
expandedName = as.character(gsub("[:space:]", "", text)),
startExpanded = FALSE,
condition = NULL
)

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

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

**Arguments**

... `menuSubItem`.

*disable*  
If TRUE, the sidebar will be disabled.

*width*  
The width of the sidebar. This must either be a number which specifies the width in pixels, or a string that specifies the width in CSS units.

*skin*  
Sidebar skin. "dark" or "light".

*status*  
Sidebar status. Valid statuses are defined as follows:

- primary: #007bff.
- secondary: #6c757d.
- info: #17a2b8.
- success: #28a745.
- warning: #ffe133.
- danger: #dc3545.
- gray-dark: #343a40.
- gray: #adb5bd.
- white: #fff.
- indigo: #6610f2.
- lightblue: #3c8dbc.
• navy: #001f3f.
• purple: #605ca8.
• fuchsia: #f012be.
• pink: #e83e8c.
• maroon: #d81b60.
• orange: #ff851b.
• lime: #01ff70.
• teal: #39cccc.
• olive: #3d9970.

elevation Sidebar elevation. 4 by default (until 5).
collapsed If TRUE, the sidebar will be collapsed on app startup.
minified Whether to slightly close the sidebar but still show item icons. Default to TRUE.
expandOnHover Whether to expand the sidebar on hover. TRUE by default.
fixed Whether to fix the sidebar. Default to TRUE.
id For sidebarMenu, if id is present, this id will be used for a Shiny input value, and it will report which tab is selected. For example, if id="tabs", then input$tabs will be the tabName of the currently-selected menuItem.
customArea Sidebar bottom space area. Only works if sidebar is fixed.
session Shiny session object.
.list An optional list containing items to put in the menu Same as the ... arguments, but in list format. This can be useful when working with programmatically generated items.
flat Whether sidebar items should have a flat design. FALSE by default.
compact Whether items should be compacted. FALSE by default.
childIndent Whether to indent children. TRUE by default.
legacy Whether to use the old adminLTE2 item selection display. Default to FALSE.
text Item name.
icon An icon tag, created by icon. If NULL, don’t display an icon.
badgeLabel A label for an optional badge. Usually a number or a short word like "new".
badgeColor A color for the badge. Valid colors:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffc107.
  • danger: #dc3545.
tabName Should correspond exactly to the tabName given in tabItem.
href An link address. Not compatible with tabName.
newTab If href is supplied, should the link open in a new browser tab?
selected If TRUE, this menuSubItem will start selected. If no item have selected=TRUE.
expandedName  A unique name given to each `menuItem` that serves to indicate which one (if any) is currently expanded. (This is only applicable to `menuItem`s that have children and it is mostly only useful for bookmarking state.)

startExpanded  Whether to expand the `menuItem` at start.

condition  When using `menuItem` with `conditionalPanel`, write the condition here (see https://github.com/RinteRface/bs4Dash/issues/35).

title  title.

name  Name of the user.

image  A filename or URL to use for an image of the person. If it is a local file, the image should be contained under the www/ subdirectory of the application.

inputId  The id of the `tabsetPanel`, `navlistPanel`, or `navbarPage` object.

**Note**

See examples for a use case of the condition parameter.

**Author(s)**

David Granjon, <dgranjon@ymail.com>

**See Also**

dashboardBody

**Examples**

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(id = "sidebar"),
      body = dashboardBody(
        actionButton(inputId = "sidebarToggle", label = "Toggle Sidebar")
      )
    ),
    server = function(input, output, session) {
      observeEvent(input$sidebar, {
        if (input$sidebar) {
          showModal(modalDialog(
            title = "Alert",
            "The sidebar is opened.",
            easyClose = TRUE,
            footer = NULL
          ))
        }
      })
    }
  )
}
observeEvent(input$sidebarToggle, {
    updateSidebar(id = "sidebar", session = session)
})

observe({
    print(input$sidebar)
})
}
}

if (interactive()) {

# sidebarItem with conditional value
library(shiny)
library(bs4Dash)

ui <- dashboardPage(
    dashboardHeader(),
    dashboardSidebar(
        sidebarMenu(
            id = "sidebarMenu",
            menuItem(
                text = "Tab 1",
                tabName = "tab1"
            ),
            menuItem(
                condition = "input.show == true",
                text = "Tab 2",
                tabName = "tab2"
            )
        )
    ),
    dashboardBody(
        tabItems(
            tabItem(
                tabName = "tab1",
                h1("Welcome!")
            ),
            tabItem(
                tabName = "tab2",
                h1("Hey! You found me!")
            )
        )
    )
)

server <- function(input, output) {}
shinyApp(ui = ui, server = server)
}

## Only run this example in interactive R sessions
if (interactive()) {
    library(shiny)
    library(bs4Dash)
shinyApp(
    ui = dashboardPage(
        header = dashboardHeader(skin = "dark"),
        body = dashboardBody(
            tabItems(
                tabItem(
                    tabName = "tab1",
                    sliderInput("obs", "Number of observations:",
                        min = 0, max = 1000, value = 500
                    ),
                    plotOutput("distPlot")
                ),
                tabItem(
                    tabName = "tab2",
                    checkboxGroupInput(
                        "variable", "Variables to show:",
                        c("Cylinders" = "cyl",
                        "Transmission" = "am",
                        "Gears" = "gear"
                    ),
                    tableOutput("data")
                ),
                tabItem(
                    tabName = "tab3",
                    checkboxInput("val", "Some value", FALSE),
                    textOutput("value")
                ),
                tabItem(
                    tabName = "tab4",
                    "Nothing special here!"
                ),
                tabItem(
                    tabName = "tab5",
                    "Tab 5"
                ),
                tabItem(
                    tabName = "tab6",
                    "Tab 6"
                ),
                tabItem(
                    tabName = "tab7",
                    "Tab 7"
                )
            ),
        sidebar = dashboardSidebar(
            skin = "light",
            inputId = "sidebarState",
            sidebarMenu(
                id = "sidebar",
                menuItem(
                    "Tab 1"
                )
            )
        ),
    )
)
text = "Tab 1",
tabName = "tab1",
icon = icon("shuttle-van")
),
menuItem(
    text = "Tab 2",
    tabName = "tab2",
    icon = icon("space-shuttle"),
    selected = TRUE
),
menuItem(
    text = "Item List 1",
    icon = icon("bars"),
    startExpanded = TRUE,
    menuSubItem(
        text = "Item 3",
        tabName = "tab3",
        icon = icon("circle-thin")
    ),
    menuSubItem(
        text = "Item 4",
        tabName = "tab4",
        icon = icon("circle-thin")
    )
),
menuItem(
    text = "Item List 2",
    icon = icon("bars"),
    startExpanded = FALSE,
    menuSubItem(
        text = "Item 5",
        tabName = "tab5",
        icon = icon("circle-thin")
    ),
    menuSubItem(
        text = "Item 6",
        tabName = "tab6",
        icon = icon("circle-thin")
    )
),
menuItem(
    text = "Tab 7",
    tabName = "tab7",
    icon = icon("home")
)
),
controlbar = dashboardControlbar(  
    skin = "light",
    sliderInput(  
        inputId = "controller",
        label = "Update the first tabset",
        min = 1,
bs4DropdownMenu

**Description**

`bs4DropdownMenu` creates an adminLTE3 dashboard dropdown menu, to be inserted in a `dashboard-Header`

`messageItem` creates a message item to place in a `dropdownMenu`

`messageItem` creates a message item to place in a `dropdownMenu`

`taskItem` creates a task item to place in a `dropdownMenu`
Usage

bs4DropdownMenu(
  ..., 
  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 = shiny::icon("exclamation-triangle"),
  status = "success",
  href = NULL,
  inputId = NULL
)

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

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

Arguments

... Items to put in the menu. Typically, message menus should contain messageItems, notification menus should contain notificationItems, and task menus should contain taskItems.

type The type of menu. Should be one of "messages", "notifications", "tasks".
badgeStatus  The status of the badge which displays the number of items in the menu. This
determines the badge’s color. Valid statuses are defined as follows:

• primary: #007bff.
• secondary: #6c757d.
• info: #17a2b8.
• success: #28a745.
• warning: #ffc107.
• danger: #dc3545.

A value of NULL means to not display a badge.

icon An icon tag, created by icon.

headerText An optional text argument used for the header of the dropdown menu (this is
only visible when the menu is expanded). If none is provided by the user, the
default is "You have x messages," where x is the number of items in the menu
(if the type is specified to be "notifications" or "tasks," the default text shows
"You have x notifications" or "You have x tasks," respectively).

list An optional list containing items to put in the menu Same as the . . . arguments,
but in list format. This can be useful when working with programmatically
generated items.

href An optional URL to link to.

from Who the message is from.

message Text of the message.

time String representing the time the message was sent. Any string may be used. For
example, it could be a relative date/time like "5 minutes", "today", or "12:30pm
yesterday", or an absolute time, like "2014-12-01 13:45". If NULL, no time will
be displayed.

image User image.

color A color for the bar. Valid colors are defined as follows:

• primary: #007bff.
• secondary: #6c757d.
• info: #17a2b8.
• success: #28a745.
• warning: #ffc107.
• danger: #dc3545.
• gray-dark: #343a40.
• gray: #adb5bd.
• primary: #007bff.
• secondary: #6c757d.
• info: #17a2b8.
• success: #28a745.
• warning: #ffc107.
• danger: #dc3545.
• gray-dark: #343a40.
• gray: #adb5bd.
• white: #fff.
• indigo: #6610f2.
• lightblue: #3c8dbc.
• navy: #001f3f.
• purple: #605ca8.
• fuchsia: #f012be.
• pink: #e83e8c.
bs4DropdownMenu

- maroon: #d81b60.
- orange: #ff851b.
- lime: #01ff70.
- teal: #39cccc.
- olive: #3d9970.

inputId
Whether to allow the item to act as a actionButton.

text
The task text.

status
The status of the item. This determines the item’s background color. Valid statuses are defined as follows:
- primary: #007bff.
- secondary: #6c757d.
- info: #17a2b8.
- success: #28a745.
- warning: #ffc107.
- danger: #dc3545.
- gray-dark: #343a40.
- gray: #adb5bd.
- white: #fff.
- indigo: #6610f2.
- lightblue: #3c8dbc.
- navy: #001f3f.
- purple: #605ca8.
- fuchsia: #f012be.
- pink: #e83e8c.
- maroon: #d81b60.
- orange: #ff851b.
- lime: #01ff70.
- teal: #39cccc.
- olive: #3d9970.

value
A percent value to use for the bar.

Author(s)
David Granjon, <dgranjon@ymail.com>

See Also
dashboardHeader for example usage.

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)
```
shinyApp(  
  ui = dashboardPage(    
    header = dashboardHeader(      
      rightUi = dropdownMenu(        
        badgeStatus = "danger",        
        type = "messages",        
        messageItem(                      
          inputId = "triggerAction1",          
          message = "message 1",          
          from = "Divad Nojnarg",          
          image = "https://adminlte.io/themes/v3/dist/img/user3-128x128.jpg",          
          time = "today",          
          color = "lime"        
        )        
      ),        
      leftUi = tagList(                      
        dropdownMenu(                        
          badgeStatus = "info",                        
          type = "notifications",                        
          notificationItem(                                      
            inputId = "triggerAction2",                                      
            text = "Error!",                                      
            status = "danger"                                      
          )                                      
        ),                        
        dropdownMenu(                        
          badgeStatus = "info",                        
          type = "tasks",                        
          taskItem(                                                      
            inputId = "triggerAction3",                                                      
            text = "My progress",                                                      
            color = "orange",                                                      
            value = 10                                                      
          )                                                      
        )                                                      
      )                        
    ),                        
    sidebar = dashboardSidebar(),                        
    controlbar = dashboardControlbar(),                        
    footer = dashboardFooter(),                        
    title = "dropdownMenu",                        
    body = dashboardBody()                        
  ),                        
  server = function(input, output) {                        
    observeEvent(input$triggerAction1, {                        
      showModal(modalDialog(                                                      
        title = "Important message",                                                      
        "This is an important message!"
      ))                        
    })                        
  })
bs4Jumbotron

Description

Create a jumbotron

Usage

```r
bs4Jumbotron(
  ...,  # Any content.
  title = NULL,
  lead = NULL,
  href = NULL,
  btnName = "More",
  status = c("primary", "warning", "danger", "info", "success")
)
```

```r
jumbotron(
  ...,  # Any content.
  title = NULL,
  lead = NULL,
  href = NULL,
  btnName = "More",
  status = c("primary", "warning", "danger", "info", "success")
)
```

Arguments

- `...`: Any content.
- `title`: Jumbotron title.
- `lead`: Jumbotron lead.
- `href`: Jumbotron external link.
- `btnName`: Jumbotron button name.
- `status`: Jumbotron background color. "primary", "success", "warning", "danger" or "info".

Author(s)

David Granjon, <dgranjon@ymail.com>
**Examples**

```r
if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Jumbotron",
      body = dashboardBody(
        jumbotron(
          title = "Hello, world!",
          lead = "This is a simple hero unit, a simple jumbotron-style component for calling extra attention to featured content or information.",
          status = "primary",
          href = "https://www.google.com"
        )
      ),
    ),
    server = function(input, output) {}
  )
}
```

---

**bs4ListGroup**  
*BS4 list group for AdminLTE3*

### Description

Create a list group

Create a list group item

### Usage

```r
bs4ListGroup(
  ...,  
  type = c("basic", "action", "heading"),
  width = 4,
  .list = NULL
)
```

```r
bs4ListGroupItem(
```
Arguments

... Item content.

**type** List group type.

**width** List group width. 4 by default. Between 1 and 12.

**.list** Slot for programmatically generated items.

**title** Item title (only if type is "heading").

**subtitle** Item subtitle (only if type is "heading").

**footer** Item footer content (only if type is "heading").

**active** Whether the item is active or not. FALSE by default. Only if type is "action" or "heading".

**disabled** Whether the item is disabled or not. FALSE by default. Only if type is "action" or "heading".

**href** Item external link.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
```
ui = dashboardPage(
    header = dashboardHeader(),
    sidebar = dashboardSidebar(),
    controlbar = dashboardControlbar(),
    footer = dashboardFooter(),
    title = "test",
    body = dashboardBody(
        fluidRow(
            listGroup(
                type = "basic",
                listGroupItem("Cras justo odio"),
                listGroupItem("Dapibus ac facilisis in"),
                listGroupItem("Morbi leo risus")
            ),
            listGroup(
                type = "action",
                listGroupItem("Cras justo odio",
                    active = TRUE,
                    disabled = FALSE,
                    href = "https://www.google.com"
                ),
                listGroupItem(
                    active = FALSE,
                    disabled = FALSE,
                    "Dapibus ac facilisis in",
                    href = "https://www.google.com"
                ),
                listGroupItem(
                    "Morbi leo risus",
                    active = FALSE,
                    disabled = TRUE,
                    href = "https://www.google.com"
                )
            ),
            listGroup(
                type = "heading",
                listGroupItem("Donec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit.".,
                    active = TRUE,
                    disabled = FALSE,
                    title = "List group item heading",
                    subtitle = "3 days ago",
                    footer = "Donec id elit non mi porta."
                ),
                listGroupItem("Donec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit.",
                    active = FALSE,
                    disabled = FALSE,
                    title = "List group item heading",
                    subtitle = "3 days ago",
                    footer = "Donec id elit non mi porta."
                )
            ),
        ))
    )
 bs4Loading

footer = "Donec id elit non mi porta."
)
)
)
)
),
server = function(input, output) {
}
}

bs4Loading AdminLTE3 loading state element

Description

When a section is still work in progress or a computation is running

Usage

bs4Loading()

loadingState()

Note

Loading state can be programmatically used when a computation is running for instance.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if(interactive()){
library(shiny)
library(bs4Dash)

shinyApp(
ui = dashboardPage(
dashboardHeader(),
dashboardSidebar(),
dashboardBody(
box(
title = "loading spinner",
loadingState()
)
),
title = "Loading State"
),


server = function(input, output) { }
}

bs4ProgressBar  AdminLTE3 progress bar

**Description**

Create a Bootstrap 4 progress bar.

**Usage**

```r
bs4ProgressBar(
  value,
  min = 0,
  max = 100,
  vertical = FALSE,
  striped = FALSE,
  animated = FALSE,
  status = "primary",
  size = NULL,
  label = NULL
)
```

```r
bs4MultiProgressBar(
  value,
  min = 0,
  max = 100,
  vertical = FALSE,
  striped = FALSE,
  animated = FALSE,
  status = "primary",
  size = NULL,
  label = NULL
)
```

```r
progressBar(
  value,
  min = 0,
  max = 100,
  vertical = FALSE,
  striped = FALSE,
  animated = FALSE,
  status = "primary",
  size = NULL,
  label = NULL
)
```
label = NULL
}

multiProgressBar(
  value,
  min = 0,
  max = 100,
  vertical = FALSE,
  striped = FALSE,
  animated = FALSE,
  status = "primary",
  size = NULL,
  label = NULL
)

Arguments

value      Progress bar value.
min        Progress bar minimum value.
max        Progress bar maximum value.
vertical   Whether to display the progress bar in vertical mode. FALSE by default.
striped    Whether the progress bar is striped or not. FALSE by default.
animated   Whether to animate the progress bar. Default to FALSE.
status     Progress bar status. Valid colors are defined as follows:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ff9800.
  • danger: #dc3545.
  • gray-dark: #343a40.
  • gray: #adb5bd.
  • white: #fff.
  • indigo: #6610f2.
  • lightblue: #3c8dbc.
  • navy: #001f3f.
  • purple: #605ca8.
  • fuchsia: #f012be.
  • pink: #e83e8c.
  • maroon: #d81b60.
  • orange: #f34e0e.
  • lime: #01ff70.
  • teal: #39cccc.
  • olive: #3d9970.
size       Progress bar size. NULL, "sm", "xs" or "xxx".
label      Progress label. NULL by default.
Details

For `multiProgressBar()`, value can be a vector which corresponds to the progress for each segment within the progress bar. If supplied, striped, animated, status, and label must be the same length as value or length 1, in which case vector recycling is used.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if(interactive()){  
  library(shiny)  
  library(bs4Dash)  

  shinyApp({  
    ui = dashboardPage({  
      header = dashboardHeader(),  
      sidebar = dashboardSidebar(),  
      body = dashboardBody({  
        box({  
          title = "Horizontal",  
          progressBar({  
            value = 10,  
            striped = TRUE,  
            animated = TRUE  
          }),  
          progressBar({  
            value = 50,  
            status = "warning",  
            size = "xs"  
          }),  
          progressBar({  
            value = 20,  
            status = "danger",  
            size = "sm"  
          }),  
          multiProgressBar({  
            value = c(50, 20),  
            status = c("warning", "danger"),  
            size = "sm"  
          })  
        }),  
        box({  
          title = "Vertical",  
          progressBar({  
            value = 10,  
            striped = TRUE,  
            animated = TRUE,  
            vertical = TRUE  
          }),  
          progressBar({  
            value = 50,  
            striped = TRUE,  
            animated = TRUE,  
            vertical = TRUE  
          }),  
          progressBar({  
            value = 20,  
            striped = TRUE,  
            animated = TRUE,  
            vertical = TRUE  
          })  
        })  
      })  
    })  
  })
```
value = 50,
status = "warning",
size = "xs",
vertical = TRUE
),
progressBar(
  value = 20,
  status = "danger",
  size = "sm",
  vertical = TRUE
),
multiProgressBar(
  value = c(50, 20),
  status = c("warning", "danger"),
  size = "sm",
  vertical = TRUE
)
)
)
),
title = "Progress bars"
),
server = function(input, output) { }
}
}

bs4Quote

**Bootstrap 4 block quote**

---

**Description**

Build a bootstrap 4 block quote

**Usage**

bs4Quote(..., color)

blockQuote(..., color)

**Arguments**

... Content.

color Block color. Valid colors are defined as follows:
  - primary: #007bff.
  - secondary: #6c757d.
  - info: #17a2b8.
  - success: #28a745.
  - warning: #ffc107.
• danger: #dc3545.
• gray-dark: #343a40.
• gray: #adb5bd.
• white: #fff.
• indigo: #6610f2.
• lightblue: #3c8dbc.
• navy: #001f3f.
• purple: #605ca8.
• fuchsia: #f012be.
• pink: #e83e8c.
• maroon: #d81b60.
• orange: #ff851b.
• lime: #01ff70.
• teal: #39ccc.
• olive: #3d9970.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        fluidRow(
          blockQuote("Blablabla", color = "indigo"),
          blockQuote("Blablabla", color = "danger"),
          blockQuote("Blablabla", color = "teal"),
          blockQuote("Blablabla", color = "orange"),
          blockQuote("Blablabla", color = "warning"),
          blockQuote("Blablabla", color = "fuchsia")
        )
      ),
      footer = dashboardFooter()
    ),
    server = function(input, output) { }
  )
}
```
bs4Ribbon  

Bootstrap 4 ribbon

Description

bs4Ribbon build a bootstrap 4 ribbon

Usage

bs4Ribbon(text, color)

ribbon(text, color)

Arguments

text   Ribbon text.
color  Ribbon color. Valid colors are defined as follows:

• primary: #007bff.
• secondary: #6c757d.
• info: #17a2b8.
• success: #28a745.
• warning: #ffc107.
• danger: #dc3545.
• gray-dark: #343a40.
• gray: #adb5bd.
• white: #fff.
• indigo: #6610f2.
• lightblue: #3c8dbc.
• navy: #001f3f.
• purple: #605ca8.
• fuchsia: #f012be.
• pink: #e83e8c.
• maroon: #d81b60.
• orange: #ff851b.
• lime: #01ff70.
• teal: #39cccc.
• olive: #3d9970.

Author(s)

David Granjon, <dgranjon@ymail.com>
Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        fluidRow(
          box(
            width = 4,
            title = "Blue ribbon",
            bs4Ribbon(
              text = "New",
              color = "primary"
            )
          ),
          box(
            width = 4,
            title = "Purple ribbon",
            bs4Ribbon(
              text = "New",
              color = "indigo"
            )
          ),
          box(
            width = 4,
            title = "Orange ribbon",
            bs4Ribbon(
              text = "New",
              color = "orange"
            )
          ),
          footer = dashboardFooter()
        ),
        server = function(input, output) { }
      )
    ),
  )
}
```
Description

socialBox Creates social card
userBlock goes in the title of socialBox.
Create a card comment to insert in socialBox

Usage

bs4SocialCard(
  ...
  title = NULL,
  footer = NULL,
  width = 6,
  height = NULL,
  collapsible = TRUE,
  collapsed = FALSE,
  closable = FALSE,
  maximizable = FALSE,
  boxToolSize = "sm",
  elevation = NULL,
  headerBorder = TRUE,
  label = NULL,
  dropdownMenu = NULL,
  sidebar = NULL,
  id = NULL
)

userBlock(image, title, subtitle = NULL)

cardComment(..., image, title = NULL, date = NULL)

socialBox(
  ...
  title = NULL,
  footer = NULL,
  width = 6,
  height = NULL,
  collapsible = TRUE,
  collapsed = FALSE,
  closable = FALSE,
  maximizable = FALSE,
  boxToolSize = "sm",
  elevation = NULL,
  headerBorder = TRUE,
  label = NULL,
  dropdownMenu = NULL,
  sidebar = NULL,
  id = NULL
)
boxComment(..., image, title = NULL, date = NULL)

Arguments

... Comment content.
title Comment title.
footer Optional footer text.
width The width of the box, using the Bootstrap grid system. This is used for row-based layouts. The overall width of a region is 12, so the default card width of 6 occupies 1/2 of that width. For column-based layouts, use NULL for the width; the width is set by the column that contains the box.
height The height of a box, in pixels or other CSS unit. By default the height scales automatically with the content.
collapsible If TRUE, display a button in the upper right that allows the user to collapse the box.
collapsed If TRUE, start collapsed. This must be used with collapsible=TRUE.
closable If TRUE, display a button in the upper right that allows the user to close the box.
maximizable If TRUE, the card can be displayed in full screen mode.
boxToolSize Size of the toolbox: choose among "xs", "sm", "md", "lg".
elevation Card elevation.
headerBorder Whether to display a border between the header and body. TRUE by default
label Slot for boxLabel.
dropdownMenu List of items in the boxtool dropdown menu. Use boxDropdown.
sidebar Slot for boxSidebar.
id Card id.
image Author image, if any.
subtitle Any subtitle.
date Date of publication.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

Other cards: bs4CardLayout(), bs4TabCard(), bs4UserCard(), descriptionBlock(), renderbs4InfoBox(), renderbs4ValueBox()

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4Timeline(), cardDropdown(), cardProfile(), descriptionBlock(), userPost()

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4Timeline(), cardDropdown(), cardProfile(), descriptionBlock(), userPost()
Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        socialBox(
          title = userBlock(
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user4-128x128.jpg",
            title = "Social Box",
            subtitle = "example-01.05.2018"
          ),
          "Some text here!",
          attachmentBlock(
            image = "https://adminlte.io/themes/v3/dist/img/user1-128x128.jpg",
            title = "Test",
            href = "https://google.com",
            "This is the content"
          ),
          lapply(X = 1:10, FUN = function(i) {
            boxComment(
              image = "https://adminlte.io/themes/AdminLTE/dist/img/user3-128x128.jpg",
              title = paste("Comment", i),
              date = "01.05.2018",
              paste0("The ", i, "-th comment")
            )
          })),
        footer = "The footer here!"
      ),
      controlbar = dashboardControlbar(),
      title = "socialBox"
    ),
    server = function(input, output) { }
  )
}
```

---

**Description**

Create a sortable UI section
bs4Sortable

Usage

bs4Sortable(..., width = 12)

sortable(..., width = 12)

Arguments

... Slot for UI elements such as box.
width Section width: between 1 and 12.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Sortable UI",
      body = dashboardBody(
        fluidRow(
          lapply(1:3, FUN = function(i) {
            sortable(
              width = 4,
              p(class = "text-center", paste("Column", i)),
              lapply(1:2, FUN = function(j) {
                box(
                  title = paste0("I am the ", j,"-th card of the ", i,"-th column"),
                  width = 12,
                  "Click on my header"
                )
              )
            )
          })
        ),
        server = function(input, output) {}
      )
    )
  )
}
bs4Stars

AdminLTE3 stars

Description

Create a block of stars (ideal for rating)

Usage

bs4Stars(value, max = 5, color = "warning")

starBlock(value, max = 5, color = "warning")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Current value. Should be positive and lower or equal to max.</td>
</tr>
<tr>
<td>max</td>
<td>Maximum number of stars by block.</td>
</tr>
<tr>
<td>color</td>
<td>Star color. Valid colors are listed below:</td>
</tr>
<tr>
<td></td>
<td>• primary: #007bff.</td>
</tr>
<tr>
<td></td>
<td>• secondary: #6c757d.</td>
</tr>
<tr>
<td></td>
<td>• info: #17a2b8.</td>
</tr>
<tr>
<td></td>
<td>• success: #28a745.</td>
</tr>
<tr>
<td></td>
<td>• warning: #ffc107.</td>
</tr>
<tr>
<td></td>
<td>• danger: #dc3545.</td>
</tr>
<tr>
<td></td>
<td>• gray-dark: #343a40.</td>
</tr>
<tr>
<td></td>
<td>• gray: #adb5bd.</td>
</tr>
<tr>
<td></td>
<td>• white: #fff.</td>
</tr>
<tr>
<td></td>
<td>• indigo: #6610f2.</td>
</tr>
<tr>
<td></td>
<td>• lightblue: #3c8dbc.</td>
</tr>
<tr>
<td></td>
<td>• navy: #001f3f.</td>
</tr>
<tr>
<td></td>
<td>• purple: #90d5cd.</td>
</tr>
<tr>
<td></td>
<td>• fuchsia: #f012be.</td>
</tr>
<tr>
<td></td>
<td>• pink: #e83e8c.</td>
</tr>
<tr>
<td></td>
<td>• maroon: #d81b60.</td>
</tr>
<tr>
<td></td>
<td>• orange: #f056b6.</td>
</tr>
<tr>
<td></td>
<td>• lime: #6f3e00.</td>
</tr>
<tr>
<td></td>
<td>• teal: #39cccc.</td>
</tr>
<tr>
<td></td>
<td>• olive: #3d9970.</td>
</tr>
</tbody>
</table>

Author(s)

David Granjon, <dgranjon@ymail.com>
Examples

```r
if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "Star example",
          starBlock(5),
          starBlock(5, color = "fuchsia"),
          starBlock(1, color = "danger"),
          starBlock(3, color = "secondary")
        ),
        title = "starBlock"
      ),
      server = function(input, output) { }
    )
  )
}
```

---

**bs4TabCard**

Create a Bootstrap 4 tabCard

---

**Description**

Build an adminLTE3 card with tabs

**Usage**

```r
bs4TabCard(
  ..., 
  id = NULL, 
  selected = NULL, 
  title = NULL, 
  width = 6, 
  height = NULL, 
  side = c("left", "right"), 
  type = NULL, 
  footer = NULL, 
  status = NULL, 
  solidHeader = FALSE, 
  background = NULL, 
  collapsible = TRUE, 
) ```
bs4TabCard

collapsed = FALSE,
closable = FALSE,
maximizable = FALSE,
icon = NULL,
gradient = FALSE,
boxToolSize = "sm",
elevation = NULL,
headerBorder = TRUE,
label = NULL,
dropdownMenu = NULL,
sidebar = NULL,
.list = NULL
)

tabBox(
  ..., 
  id = NULL,
  selected = NULL,
  title = NULL,
  width = 6,
  height = NULL,
  side = c("left", "right"),
  type = NULL,
  footer = NULL,
  status = NULL,
  solidHeader = FALSE,
  background = NULL,
collapsible = TRUE,
collapsed = FALSE,
closable = FALSE,
maximizable = FALSE,
icon = NULL,
gradient = FALSE,
boxToolSize = "sm",
elevation = NULL,
headerBorder = TRUE,
label = NULL,
dropdownMenu = NULL,
sidebar = NULL,
.list = NULL
)

**Arguments**

...  Contents of the box.
id  Card id.
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.
**bs4TabCard**

- **title**: Optional title.
- **width**: The width of the box, using the Bootstrap grid system. This is used for row-based layouts. The overall width of a region is 12, so the default card width of 6 occupies 1/2 of that width. For column-based layouts, use NULL for the width; the width is set by the column that contains the box.
- **height**: The height of a box, in pixels or other CSS unit. By default, the height scales automatically with the content.
- **side**: `tabPanel` side. Either left or right.
- **type**: `"tabs"` Standard tab look
  - `"pills"` Selected tabs use the background fill color
- **footer**: Optional footer text.
- **status**: The status of the item. This determines the item’s background color. Valid statuses are defined as follows:
  - primary: #007bff.
  - secondary: #6c757d.
  - info: #17a2b8.
  - success: #28a745.
  - warning: #ffc107.
  - danger: #dc3545.
  - gray-dark: #343a40.
  - gray: #adb5bd.
  - white: #fff.
  - indigo: #6610f2.
  - lightblue: #3c8dbc.
  - navy: #001f3f.
  - purple: #605ca8.
  - fuchsia: #f012be.
  - pink: #e83e8c.
  - maroon: #d81b60.
  - orange: #ff851b.
  - lime: #01ff70.
  - teal: #39cccc.
  - olive: #3d9970.
- **solidHeader**: Should the header be shown with a solid color background?
- **background**: If NULL (the default), the background of the box will be white. Otherwise, a color string. Valid colors are listed in validColors. See below:
  - primary: #007bff.
  - secondary: #6c757d.
  - info: #17a2b8.
  - success: #28a745.
  - warning: #ffc107.
bs4TabCard

- danger: #dc3545.
- gray-dark: #343a40.
- gray: #adb5bd.
- white: #fff.
- indigo: #6610f2.
- lightblue: #3c8dbc.
- navy: #001f3f.
- purple: #605ca8.
- fuchsia: #f012be.
- pink: #e83e8c.
- maroon: #d81b60.
- orange: #ff851b.
- lime: #01ff70.
- teal: #39cccc.
- olive: #3d9970.

collapsible If TRUE, display a button in the upper right that allows the user to collapse the box.
collapsed If TRUE, start collapsed. This must be used with collapsible=TRUE.
closable If TRUE, display a button in the upper right that allows the user to close the box.
maximizable If TRUE, the card can be displayed in full screen mode.
icon Header icon. Displayed before title. Expect icon.
gradients Whether to allow gradient effect for the background color. Default to FALSE.
boxToolSize Size of the toolbox: choose among "xs", "sm", "md", "lg".
elevation Card elevation.
headerBorder Whether to display a border between the header and body. TRUE by default.
label Slot for boxLabel.
dropdownMenu List of items in the boxtool dropdown menu. Use boxDropdown.
sidebar Slot for boxSidebar.
.list In case of programmatically generated items. See example.

Note
User will access the tabBox input with input$<id>_box. This allows to get the state of the box and update it on the server with updateBox.

Author(s)
David Granjon, <dgranjon@ymail.com>

See Also
Other cards: bs4CardLayout(), bs4SocialCard(), bs4UserCard(), descriptionBlock(), renderbs4InfoBox(), renderbs4ValueBox()
Examples

if (interactive()) {
  library(shiny)
  library(bs4Dash)

  menu_tab <- lapply(1:3, function(i) {
    tabPanel(
      sprintf("Menu %s", i),
      sprintf("Hello tab %s", i)
    )
  })

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "tabBox",
      body = dashboardBody(
        tabBox(
          id = "tabcard",
          title = "A card with tabs",
          selected = "Tab 2",
          status = "primary",
          solidHeader = FALSE,
          type = "tabs",
          tabPanel(
            title = "Tab 1",
            "Content 1"
          ),
          tabPanel(
            title = "Tab 2",
            "Content 2"
          ),
          tabPanel(
            title = "Tab 3",
            "Content 3"
          )
        ),
        tabBox(
          id = "mybox2",
          title = "",
          .list = menu_tab
        ),
        selectInput(
          "tab",
          "Selected a tab",
          choices = paste("Menu", 1:3),
          "Menu 2"
        )
      )
  )
}
bs4Timeline

server = function(input, output, session) {
  observeEvent(input$tab, {
    updateTabsetPanel(session, inputId = "mybox2", input$tab)
  })
}
}

bs4Timeline  AdminLTE3 timeline block

Description

timelineBlock creates a timeline block that may be inserted in a box or outside.
timelineLabel creates a timeline label element to highlight an event.
timelineItem creates a timeline item that contains information for a given event like the title, description, date, ...
timelineItemMedia create a specific container for images.
timelineStart indicates a starting point.
timelineEnd indicates an end point.

Usage

ds4Timeline(..., reversed = TRUE, width = 6)

ds4TimelineLabel(..., color = NULL)

ds4TimelineItem(
  ..., 
  icon = NULL, 
  color = NULL, 
  time = NULL, 
  title = NULL, 
  border = TRUE, 
  footer = NULL, 
  elevation = NULL
)

ds4TimelineItemMedia(image = NULL, height = NULL, width = NULL)

ds4TimelineStart(icon = shiny::icon("clock"), color = NULL)

ds4TimelineEnd(icon = shiny::icon("hourglass-end"), color = NULL)

timelineBlock(..., reversed = TRUE, width = 6)
timelineLabel(..., color = NULL)

timelineItem(
    ..., 
    icon = NULL, 
    color = NULL, 
    time = NULL, 
    title = NULL, 
    border = TRUE, 
    footer = NULL, 
    elevation = NULL
)

timelineItemMedia(image = NULL, height = NULL, width = NULL)

timelineStart(icon = shiny::icon("clock"), color = NULL)

timelineEnd(icon = shiny::icon("hourglass-end"), color = NULL)

Arguments

... Any element such as timelineItemMedia ...
reversed Whether the timeline is reversed or not.
width Media width in pixels.
color Item color. Valid colors are defined as follows:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffc107.
  • danger: #dc3545.
  • gray-dark: #343a40.
  • gray: #adb5bd.
  • white: #fff.
  • indigo: #6610f2.
  • lightblue: #3c8dbc.
  • navy: #001f3f.
  • purple: #605ca8.
  • fuchsia: #f012be.
  • pink: #e83e8c.
  • maroon: #d81b60.
  • orange: #ff851b.
  • lime: #01ff70.
  • teal: #01cccd.
• olive: #3d9970.

icon Item icon such as "clock", "times", ...
time Item date or time.
title Item title.
border Whether to display a border between the header and the body. TRUE by default.
footer Item footer if any.
elevation Timeline elevation (numeric). NULL by default.
image Media url or path.
height Media height in pixels.

Note

reversed is useful when the user wants to use the timeline inside a box.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), cardDropdown(), cardProfile(), descriptionBlock(), userPost()

Examples

```r
if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = bs4DashPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "test",
      body = dashboardBody(
        box(
          title = "Timeline",
          timelineBlock(
            width = 12,
            reversed = TRUE,
            timelineEnd(color = "danger"),
            timelineLabel("10 Feb. 2014", color = "pink"),
            timelineItem(
              elevation = 4,
              title = "Item 1",
              icon = icon("gears"),
              color = "olive",
```
time = "now",
footer = "Here is the footer",
"This is the body"
),
timelineItem(
title = "Item 2",
border = FALSE
),
timelineLabel("3 Jan. 2014", color = "lightblue"),
timelineItem(
elevation = 2,
title = "Item 3",
icon = icon("paint-brush"),
status = "orange",
timelineItemMedia(image = "https://via.placeholder.com/150x100"),
timelineItemMedia(image = "https://via.placeholder.com/150x100")
),
timelineStart(color = "secondary")
)
)
),
server = function(input, output) {}
)

bs4UserCard

**AdminLTE3 widget user card**

### Description

`userBox` creates a user card.

`userDescription` creates a customized title tag for `userBox`.

### Usage

```r
bs4UserCard(
  ..., 
  title = NULL, 
  footer = NULL, 
  status = NULL, 
  background = NULL, 
  width = 6, 
  height = NULL, 
  collapsible = TRUE, 
  collapsed = FALSE, 
  closable = FALSE, 
  maximizable = FALSE, 
  ```
bs4UserCard

  gradient = FALSE,
  boxToolSize = "sm",
  elevation = NULL,
  headerBorder = TRUE,
  label = NULL,
  dropdownMenu = NULL,
  sidebar = NULL,
  id = NULL
)

bs4UserDescription(
  title,
  subtitle = NULL,
  image,
  backgroundImage = NULL,
  type = c(1, 2),
  imageElevation = NULL
)

userBox(
  ..., 
  title = NULL,
  footer = NULL,
  status = NULL,
  background = NULL,
  width = 6,
  height = NULL,
  collapsible = TRUE,
  collapsed = FALSE,
  closable = FALSE,
  maximizable = FALSE,
  gradient = FALSE,
  boxToolSize = "sm",
  elevation = NULL,
  headerBorder = TRUE,
  label = NULL,
  dropdownMenu = NULL,
  sidebar = NULL,
  id = NULL
)

userDescription(
  title,
  subtitle = NULL,
  image,
  backgroundImage = NULL,
  type = c(1, 2),
  imageElevation = NULL
Arguments

... Contents of the box.
title User card title.
footer Optional footer text.
status The status of the item. This determines the item’s background color. Valid statuses are defined as follows:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffeb34.
  • danger: #dc3545.
  • gray-dark: #343a40.
  • gray: #adb5bd.
  • white: #fff.
  • indigo: #6610f2.
  • lightblue: #3c8dbc.
  • navy: #001f3f.
  • purple: #605ca8.
  • fuchsia: #ff00ff.
  • pink: #ff69b4.
  • maroon: #d85315.
  • orange: #ff8c00.
  • lime: #00ff00.
  • teal: #20c997.
  • olive: #808000.
background If NULL (the default), the background of the box will be white. Otherwise, a color string. Valid colors are listed in validColors. See below:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffeb34.
  • danger: #dc3545.
  • gray-dark: #343a40.
  • gray: #adb5bd.
  • white: #fff.
  • indigo: #6610f2.
  • lightblue: #3c8dbc.
bs4UserCard

- navy: #001f3f.
- purple: #605ca8.
- fuchsia: #f012be.
- pink: #e83e8c.
- maroon: #d81b60.
- orange: #ff851b.
- lime: #01ff70.
- teal: #39ccc.
- olive: #3d9970.

width
The width of the box, using the Bootstrap grid system. This is used for row-based layouts. The overall width of a region is 12, so the default card width of 6 occupies 1/2 of that width. For column-based layouts, use NULL for the width; the width is set by the column that contains the box.

height
The height of a box, in pixels or other CSS unit. By default the height scales automatically with the content.

collapsible
If TRUE, display a button in the upper right that allows the user to collapse the box.

collapsed
If TRUE, start collapsed. This must be used with collapsible=TRUE.

closable
If TRUE, display a button in the upper right that allows the user to close the box.

maximizable
If TRUE, the card can be displayed in full screen mode.

gradient
Whether to allow gradient effect for the background color. Default to FALSE.

boxToolSize
Size of the toolbox: choose among "xs", "sm", "md", "lg".

elevation
Card elevation.

headerBorder
Whether to display a border between the header and body. TRUE by default.

label
Slot for boxLabel.

dropdownMenu
List of items in the boxtool dropdown menu. Use boxDropdown.

sidebar
Slot for boxSidebar.

id
Card id.

subtitle
User card subtitle.

image
User image url or path.

backgroundImage
image url, if any. Background needs to be TRUE.

type
User card type. Either 1 or 2. 1 corresponds to a centered user image, while 2 is a left aligned user image.

imageElevation
User card image elevation (numeric). NULL by default.

Author(s)

David Granjon, <dgranjon@ymail.com>
See Also

Other cards: bs4CardLayout(), bs4SocialCard(), bs4TabCard(), descriptionBlock(), renderbs4InfoBox(), renderbs4ValueBox()

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "test",
      body = dashboardBody(
        userBox(
          title = userDescription(
            title = "Nadia Carmichael",
            subtitle = "lead Developer",
            type = 2,
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user7-128x128.jpg",
          ),
          status = "primary",
          gradient = TRUE,
          background = "primary",
          boxToolSize = "xl",
          "Some text here!",
          footer = "The footer here!"
        ),
        userBox(
          title = userDescription(
            title = "Alexander Pierce",
            subtitle = "Founder & CEO",
            type = 1,
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user1-128x128.jpg",
          ),
          status = "indigo",
          closable = TRUE,
          "Some text here!",
          footer = "The footer here!"
        ),
        userBox(
          title = userDescription(
            title = "Elizabeth Pierce",
            subtitle = "Web Designer",
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user3-128x128.jpg",
          ),
          status = "olive",
```
bs4UserMenu

Bootstrap 4 user profile.

Description

dashboardUser to insert in the rightUi or leftUi slot of dashboardHeader. This can be inserted in a dashboardUser. This can be used as a placeholder for dynamically-generated dashboardUser.

Usage

bs4UserMenu(
  ..., 
  name = NULL, 
  image = NULL, 
  title = NULL, 
  subtitle = NULL, 
  footer = NULL, 
  status = NULL 
)

dashboardUserItem(item, width)

userOutput(id, tag = shiny::tags$li)

renderUser(expr, env = parent.frame(), quoted = FALSE, outputArgs = list())

dashboardUser(
  ..., 
  name = NULL, 
  image = NULL, 
  title = NULL, 
  subtitle = NULL, 
  footer = NULL, 
  status = NULL 
)
Arguments

Body content. Slot for `dashboardUserItem`.

- **name**  
  User name.

- **image**  
  User profile picture.

- **title**  
  A title.

- **subtitle**  
  A subtitle.

- **footer**  
  Footer is any.

- **status**  
  Ribbon status. Valid colors are defined as follows:
  - primary: #007bff.
  - secondary: #6c757d.
  - info: #17a2b8.
  - success: #28a745.
  - warning: #ffc107.
  - danger: #dc3545.
  - gray-dark: #343a40.
  - gray: #adb5bd.
  - white: #fff.
  - indigo: #6610f2.
  - lightblue: #3c8dbc.
  - navy: #001f3f.
  - purple: #605ca8.
  - fuchsia: #f012be.
  - pink: #e83e8c.
  - maroon: #d81b60.
  - orange: #ff851b.
  - lime: #01ff70.
  - teal: #39cccc.
  - olive: #3d9970.

- **item**  
  HTML Tag.

- **width**  
  Item width between 1 and 12.

- **id**  
  Output variable name.

- **tag**  
  A tag function, like `tags$li` or `tags$ul`.

- **expr**  
  An expression that returns a Shiny tag object, `HTML()`, or a list of such objects.

- **env**  
  The parent environment for the reactive expression. By default, this is the calling environment, the same as when defining an ordinary non-reactive expression. If `expr` is a quosure and quoted is `TRUE`, then `env` is ignored.

- **quoted**  
  If it is `TRUE`, then the quoted value of `expr` will be used when `expr` is evaluated. If `expr` is a quosure and you would like to use its expression as a value for `expr`, then you must set quoted to `TRUE`.

- **outputArgs**  
  A list of arguments to be passed through to the implicit call to `uiOutput()` when `renderUI` is used in an interactive R Markdown document.
See Also

renderUser for the corresponding server side function and examples.
userOutput for the corresponding client side function and examples.

Examples

if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(rightUi = userOutput("user")),
      sidebar = dashboardSidebar(),
      body = dashboardBody(),
      title = "DashboardPage"
    ),
    server = function(input, output) {
      output$user <- renderUser(
        dashboardUser(
          name = "Divad Nojnarg",
          image = "https://adminlte.io/themes/AdminLTE/dist/img/user2-160x160.jpg",
          title = "shinydashboardPlus",
          subtitle = "Author",
          footer = p("The footer", class = "text-center"),
          fluidRow(
            dashboardUserItem(
              width = 6,
              "Item 1"
            ),
            dashboardUserItem(
              width = 6,
              "Item 2"
            )
          )
        )
      )
    }
  )
}

---

cardDropdown Create a box dropdown item list

Description

Can be used to add dropdown items to a cardtool.
To insert in boxDropdown.
Usage

    cardDropdown(..., icon = shiny::icon("wrench"));
    cardDropdownItem(..., id = NULL, href = NULL, icon = NULL);
    boxDropdown(..., icon = shiny::icon("wrench"));
    boxDropdownItem(..., id = NULL, href = NULL, icon = NULL);

Arguments

...  Item content.
icon  Optional icon. Expect icon.
id    If passed, the item will behave like an action button.
href  Target url or page.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardProfile(), descriptionBlock(), userPost()

cardProfile  AdminLTE3 card profile

description

boxProfile goes inside a box. Displays user informations in an elegant container.

Create card profile item

Usage

    cardProfile(..., image = NULL, title, subtitle = NULL, bordered = FALSE);
    cardProfileItem(title, description);
    boxProfile(..., image = NULL, title, subtitle = NULL, bordered = FALSE);
    boxProfileItem(title, description);
Arguments

... Any element such as boxProfileItem.
image Profile image, if any.
title Item title.
subtitle Subtitle.
bordered Whether the container should have a border or not. FALSE by default.
description Item info.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardDropdown(), descriptionBlock(), userPost()

Examples

if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(
        sidebarMenu(
          sidebarHeader("Main content"),
          menuItem(
            "Profile Card",
            tabName = "profile_card",
            icon = icon("desktop")
          )
        )
      ),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "boxProfile",
      body = dashboardBody(
        tabItems(
          tabItem(
            tabName = "profile_card",
            bs4Card(
              status = "primary",
              solidHeader = TRUE,
              boxProfile(
                image = "https://adminlte.io/themes/AdminLTE/dist/img/user4-128x128.jpg",
                title = "Nina Mcintire",
                subtitle = "Software Engineer",
              )
            )
          )
        )
      )
    )
  )
}
column

Boostrap 4 column system

Description

This function overwrites that of Shiny since there are differences between the Bootstrap 3 and Bootstrap 4 grid systems

Usage

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

Arguments

width

The grid width of the column (must be between 1 and 12.

... 

Elements to include within the column.

offset

The number of columns to offset this column from the end of the previous column.

Examples

if (interactive()) {
library(shiny)
library(bs4Dash)

ui <- bs4DashPage(

bordered = TRUE,
boxProfileItem(
    title = "Followers",
    description = 1322
),
boxProfileItem(
    title = "Following",
    description = 543
),
boxProfileItem(
    title = "Friends",
    description = 13287
)
)

server = function(input, output) {}
}
navbar = bs4DashNavbar(),
sidebar = dashboardSidebar(
    bs4SidebarMenu(
        bs4SidebarMenuItem(
            "Welcome!",
            tabName = "tab_welcome",
            icon = "home"
            )
    )
),

body = bs4DashBody(
    bs4TabItems(
        bs4TabItem(
            tabName = "tab_welcome",
            fluidRow(
                column(
                    width = 1,
                    offset = 11,
                    actionButton(
                        "mybutton", label = "", icon = icon("question-circle")
                        )
                )
            ),
            fluidRow(
                h2("Placeholder")
                )
            )
        )
    )
)

server <- function(input, output, session) {}
shinyApp(ui = ui, server = server)
}

createAlert  

Create a Bootstrap 4 alert on the server side

Description

createAlert creates an alert and inserts it in the DOM.
closeAlert closes an alert created via createAlert.

Usage

createAlert(  
    id = NULL,
    selector = NULL,
,...
createAlert

```r
options,
  session = shiny::getDefaultReactiveDomain()
)

closeAlert(id, session = shiny::getDefaultReactiveDomain())
```

**Arguments**

- `id`  
  Anchor id.

- `selector`  
  jQuery selector. Allow more customization for the anchor (nested tags).

- `options`  
  List of options to pass to the alert. See below:
  - `content`: Alert content.
  - `title`: Alert title.
  - `closable`: Whether to allow the user to close the alert. FALSE by default.
  - `width`: Alert width. Between 1 and 12.
  - `elevation`: Alert elevation.
  - `status`: Alert status. "primary", "success", "warning", "danger" or "info".

- `session`  
  Shiny session object.

**Note**

Unlike shinyBS, there is no need to specify an anchorId and an alertId. `id` refers to the anchorId, and the alertId is simply "anchorId-alert". On the server side, one can access the alert status by `input$<id>`. If TRUE, the alert has been created and is visible, if FALSE the alert has just been closed.

**Examples**

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        tooltip(  
          sliderInput("obs", "Observations:", 10, min = 1, max = 100),
          placement = "right",
          title = "Set me higher than 50!"
        ),  
        div(id = "myalert", style = "position: absolute; bottom: 0; right: 0;"
        ),  
        controlbar = dashboardControlbar(),
        title = "Alerts",
      ),
    ),
    server = function(input, output, session) {
      observeEvent(input$obs, {
```

```r
```
if (input$obs > 50) {
    createAlert(
        id = "myalert",
        options = list(
            title = "Alert",
            closable = TRUE,
            width = 12,
            elevations = 4,
            status = "primary",
            content = "Alert content ..."
        )
    )
} else {
    closeAlert(id = "myalert")
}

observe(print(input$myalert))

observeEvent(input$myalert, {
    alertStatus <- if (input$myalert) "opened" else "closed"
    toastColor <- if (input$myalert) "bg-lime" else "bg-fuchsia"
    toast(
        title = sprintf("Alert succesfully %s!", alertStatus),
        options = list(
            class = toastColor,
            autohide = TRUE,
            position = "topRight"
        )
    )
})
numberColor = NULL,  
numberIcon = NULL,  
header = NULL,  
text = NULL,  
rightBorder = TRUE,  
marginBottom = FALSE  
)

cardPad(..., color = NULL, style = NULL)

bs4Card(
  ...,  
title = NULL,  
footer = NULL,  
status = NULL,  
solidHeader = FALSE,  
background = NULL,  
width = 6,  
height = NULL,  
collapsible = TRUE,  
collapsed = FALSE,  
closable = FALSE,  
maximizable = FALSE,  
icon = NULL,  
gradient = FALSE,  
boxToolSize = "sm",  
elevation = NULL,  
headerBorder = TRUE,  
label = NULL,  
dropdownMenu = NULL,  
sidebar = NULL,  
id = NULL  
)

updatebs4Card(
  id,  
  action = c("remove", "toggle", "toggleMaximize", "restore", "update"),  
  options = NULL,  
  session = shiny::getDefaultReactiveDomain()  
)

box(
  ...,  
title = NULL,  
footer = NULL,  
status = NULL,  
solidHeader = FALSE,  
background = NULL,  
series = NULL  
)
width = 6,
height = NULL,
collapsible = TRUE,
collapsed = FALSE,
closable = FALSE,
maximizable = FALSE,
icon = NULL,
gradient = FALSE,
boxToolSize = "sm",
elevation = NULL,
headerBorder = TRUE,
label = NULL,
dropdownMenu = NULL,
sidebar = NULL,
id = NULL
)

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

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

boxPad(..., color = NULL, style = NULL)

**Arguments**

<table>
<thead>
<tr>
<th>number</th>
<th>Any number.</th>
</tr>
</thead>
<tbody>
<tr>
<td>numberColor</td>
<td>Number color. Valid colors are defined as follows:</td>
</tr>
<tr>
<td></td>
<td>primary: #007bff.</td>
</tr>
<tr>
<td></td>
<td>secondary: #6c757d.</td>
</tr>
<tr>
<td></td>
<td>info: #17a2b8.</td>
</tr>
<tr>
<td></td>
<td>success: #28a745.</td>
</tr>
<tr>
<td></td>
<td>warning: #fffc107.</td>
</tr>
<tr>
<td></td>
<td>danger: #dc3545.</td>
</tr>
<tr>
<td></td>
<td>gray-dark: #343a40.</td>
</tr>
<tr>
<td></td>
<td>gray: #adb5bd.</td>
</tr>
<tr>
<td></td>
<td>white: #fff.</td>
</tr>
<tr>
<td></td>
<td>indigo: #6610f2.</td>
</tr>
</tbody>
</table>
- lightblue: #3c8dbc.
- navy: #001f3f.
- purple: #605ca8.
- fuchsia: #f012be.
- pink: #e83e8c.
- maroon: #d81b60.
- orange: #ff851b.
- lime: #01ff70.
- teal: #39ccc.
- olive: #3d9970.

numberIcon Number icon, if any. Expect icon.
header Bold text.
text Additional text.
rightBorder TRUE by default. Whether to display a right border to separate two blocks. The last block on the right should not have a right border.
marginBottom FALSE by default. Set it to TRUE when the descriptionBlock is used in a box
Pad context.
... Contents of the box.

color Background color. Valid colors are defined as follows:
- primary: #007bff.
- secondary: #6c757d.
- info: #17a2b8.
- success: #28a745.
- warning: #f5ca5a.
- danger: #dc3545.
- gray-dark: #343a40.
- gray: #adb5bd.
- white: #fff.
- indigo: #6610f2.
- lightblue: #3c8dbc.
- navy: #001f3f.
- purple: #605ca8.
- fuchsia: #f012be.
- pink: #e83e8c.
- maroon: #d81b60.
- orange: #ff851b.
- lime: #01ff70.
- teal: #39ccc.
- olive: #3d9970.

style Custom CSS, if any.
title Optional title.
| footer | Optional footer text. |
| status | The status of the item. This determines the item's background color. Valid statuses are defined as follows: |
|        | - primary: #007bff. |
|        | - secondary: #6c757d. |
|        | - info: #17a2b8. |
|        | - success: #28a745. |
|        | - warning: #ff851b. |
|        | - danger: #dc3545. |
|        | - gray-dark: #343a40. |
|        | - gray: #adb5bd. |
|        | - white: #fff. |
|        | - indigo: #6610f2. |
|        | - lightblue: #3c8dbc. |
|        | - navy: #001f3f. |
|        | - purple: #605ca8. |
|        | - fuchsia: #f012be. |
|        | - pink: #e83e8c. |
|        | - maroon: #d81b60. |
|        | - orange: #ff851b. |
|        | - lime: #01ff70. |
|        | - teal: #39cccc. |
|        | - olive: #3d9970. |

| solidHeader | Should the header be shown with a solid color background? |
| background  | If NULL (the default), the background of the box will be white. Otherwise, a color string. Valid colors are listed in validColors. See below: |
|            | - primary: #007bff. |
|            | - secondary: #6c757d. |
|            | - info: #17a2b8. |
|            | - success: #28a745. |
|            | - warning: #ff851b. |
|            | - danger: #dc3545. |
|            | - gray-dark: #343a40. |
|            | - gray: #adb5bd. |
|            | - white: #fff. |
|            | - indigo: #6610f2. |
|            | - lightblue: #3c8dbc. |
|            | - navy: #001f3f. |
|            | - purple: #605ca8. |
|            | - fuchsia: #f012be. |
|            | - pink: #e83e8c. |
|            | - maroon: #d81b60. |
orange: #ff851b.
• lime: #01ff70.
• teal: #39cccc.
• olive: #3d9970.

width
The width of the box, using the Bootstrap grid system. This is used for row-based layouts. The overall width of a region is 12, so the default card width of 6 occupies 1/2 of that width. For column-based layouts, use NULL for the width; the width is set by the column that contains the box.

height
The height of a box, in pixels or other CSS unit. By default the height scales automatically with the content.

collapsible
If TRUE, display a button in the upper right that allows the user to collapse the box.

collapsed
If TRUE, start collapsed. This must be used with collapsible=TRUE.

closable
If TRUE, display a button in the upper right that allows the user to close the box.

maximizable
If TRUE, the card can be displayed in full screen mode.

icon
Header icon. Displayed before title. Expect icon.

gradient
Whether to allow gradient effect for the background color. Default to FALSE.

boxToolSize
Size of the toolbox: choose among "xs", "sm", "md", "lg".

elevation
Card elevation.

headerBorder
Whether to display a border between the header and body. TRUE by default

label
Slot for boxLabel.

dropdownMenu
List of items in the boxtool dropdown menu. Use boxDropdown.

sidebar
Slot for boxSidebar.

id
Card id.

action
Action to trigger: c("remove","toggle","toggleMaximize","restore","update").

options
If action is update, a list of new options to configure the box, such as list(title = "new title",status = NULL,solidHeader = FALSE,background = "red",width = 6,height = "200px",collapsible = FALSE,closable = FALSE).

session
Shiny session.

Author(s)
David Granjon, <dgranjon@ymail.com>
David Granjon, <dgranjon@ymail.com>

See Also
Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardDropdown(), cardProfile(), userPost()

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardDropdown(), cardProfile(), userPost()

Other cards: bs4CardLayout(), bs4SocialCard(), bs4TabCard(), bs4UserCard(), renderbs4InfoBox(), renderbs4ValueBox()
Examples

```r
# Box with descriptionBlock
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          solidHeader = FALSE,
          title = "Status summary",
          background = NULL,
          width = 4,
          status = "danger",
          footer = fluidRow(
            column(
              width = 6,
              descriptionBlock(
                number = "17%",
                numberColor = "pink",
                numberIcon = icon("caret-up"),
                header = "$35,210.43",
                text = "TOTAL REVENUE",
                rightBorder = TRUE,
                marginBottom = FALSE
              )
            ),
            column(
              width = 6,
              descriptionBlock(
                number = "18%",
                numberColor = "secondary",
                numberIcon = icon("caret-down"),
                header = "1200",
                text = "GOAL COMPLETION",
                rightBorder = FALSE,
                marginBottom = FALSE
              )
            )
          )
        ),
        title = "Description Blocks"
      ),
      server = function(input, output) { }
    )
  }
}
if (interactive()) {
```
library(shiny)
library(bs4Dash)

shinyApp(
  ui = dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
      box(
        title = "Box with right pad",
        status = "warning",
        fluidRow(
          column(width = 6),
          column(
            width = 6,
            boxPad(
              color = "purple",
              descriptionBlock(
                header = "8390",
                text = "VISITS",
                rightBorder = FALSE,
                marginBottom = TRUE
              ),
              descriptionBlock(
                header = "30%",
                text = "REFERRALS",
                rightBorder = FALSE,
                marginBottom = TRUE
              ),
              descriptionBlock(
                header = "70%",
                text = "ORGANIC",
                rightBorder = FALSE,
                marginBottom = FALSE
              )
            )
          )
        )
      ),
      title = "boxPad"
    ),
    server = function(input, output) { }
  )
)

# A box with label, sidebar, dropdown menu
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(

dashboardHeader(),
dashboardSidebar(),
dashboardBody(
  box(
    title = "Closable Box with dropdown",
    closable = TRUE,
    width = 12,
    status = "warning",
    solidHeader = FALSE,
    collapsible = TRUE,
    label = boxLabel(
      text = 1,
      status = "danger"
    ),
    dropdownMenu = boxDropdown(
      boxDropdownItem("Link to google", href = "https://www.google.com"),
      boxDropdownItem("item 2", href = "#"),
      dropdownDivider(),
      boxDropdownItem("item 3", href = "#", icon = icon("th"))
    ),
    sidebar = boxSidebar(
      startOpen = TRUE,
      id = "mycardsidebar",
      sliderInput(
        "obs",
        "Number of observations:",
        min = 0,
        max = 1000,
        value = 500
      )
    ),
    plotOutput("distPlot")
  )
),
server = function(input, output) {
  output$distPlot <- renderPlot({
    hist(rnorm(input$obs))
  })
}

# Toggle a box on the client
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  ui <- dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
      tags$style("body { background-color: ghostwhite}")
    )
  )
server <- function(input, output, session) {
  output$plot <- renderPlot({
    req(!input$mybox$collapsed)
    plot(rnorm(200))
  })

  output$box_state <- renderText({
    state <- if (input$mybox$collapsed) "collapsed" else "uncollapsed"
    paste("My box is", state)
  })

  observeEvent(input$toggle_box, {
    updateBox("mybox", action = "toggle")
  })

  observeEvent(input$remove_box, {
    updateBox("mybox", action = "remove")
  })

  observeEvent(input$restore_box, {
    updateBox("mybox", action = "restore")
  })

  observeEvent(input$mybox$visible, {
    collapsed <- if (input$mybox$collapsed) "collapsed" else "uncollapsed"
    visible <- if (input$mybox$visible) "visible" else "hidden"
    message <- paste("My box is", collapsed, "and", visible)
    showNotification(message, type = "warning", duration = 1)
  })

  observeEvent(input$update_box, {
  })
}
updateBox(
   "mybox",
   action = "update",
   options = list(
      title = h2("hello", dashboardBadge(1, color = "primary")),
      status = "warning",
      solidHeader = TRUE,
      width = 12,
      background = NULL,
      height = "900px",
      closable = FALSE
   )
)
}

observeEvent(input$update_box2, {
   updateBox(
      "mybox",
      action = "update",
      options = list(
         status = NULL,
         solidHeader = FALSE,
         width = 4,
         background = "green",
         height = "500px",
         closable = TRUE
      )
   )
})

shinyApp(ui, server)

---

dropdownDivider Creator a box dropdown divider

Description

Create a box dropdown divider

Usage

dropdownDivider()

Note

Useful to separate 2 sections of dropdown items.
Author(s)

David Granjon, <dgranjon@ymail.com>

dropdownMenuOutput  Create a dropdown menu output (client side)

Description

This is the UI-side function for creating a dynamic dropdown menu.

Usage

dropdownMenuOutput(outputId)

Arguments

outputId  Output variable name.

See Also

renderMenu for the corresponding server-side function and examples, and dropdownMenu for the corresponding function for generating static menus.

Other menu outputs: menuItemOutput(), menuOutput(), renderMenu(), sidebarMenuOutput()

getAdminLTEColors  Get all AdminLTE colors.

Description

Get all AdminLTE colors.

Usage

gGetAdminLTEColors()
**Insert a tabPanel in a tabsetPanel**

**Description**

Insert a `tabPanel` in a `tabsetPanel`

**Usage**

```r
insertTab(
  inputId, 
  tab, 
  target, 
  position = c("before", "after"), 
  select = FALSE, 
  session = shiny::getDefaultReactiveDomain()
)
```

**Arguments**

- **inputId**: `tabsetPanel` id.
- **tab**: `tabPanel` to insert.
- **target**: `tabPanel` after of before which the new tab will be inserted.
- **position**: Insert before or after: c("before","after").
- **select**: Whether to select the newly inserted tab. FALSE by default.
- **session**: Shiny session object.

**Examples**

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Handle tabs",
      body = dashboardBody(
        actionButton("add", "Add 'Dynamic' tab"),
        actionButton("remove", "Remove 'Foo' tab"),
        actionButton("hideTab", "Hide 'Foo' tab"),
        actionButton("showTab", "Show 'Foo' tab"),
        br(), br(),
        tabBox(
```
id = "tabs",
title = "A card with tabs",
selected = "Bar",
status = "primary",
solidHeader = FALSE,
type = "tabs",
tabPanel("Hello", "This is the hello tab"),
tabPanel("Foo", "This is the foo tab"),
tabPanel("Bar", "This is the bar tab")
)
)
server = function(input, output, session) {
  observeEvent(input$add, {
    insertTab(
      inputId = "tabs",
      tabPanel("Dynamic", "This a dynamically-added tab"),
      target = "Bar",
      select = TRUE
    )
  )
  }
  observeEvent(input$remove, {
    removeTab(inputId = "tabs", target = "Foo")
  )
  observeEvent(input$hideTab, {
    hideTab(inputId = "tabs", target = "Foo")
  )
  observeEvent(input$showTab, {
    showTab(inputId = "tabs", target = "Foo")
  )
}
}

ionicon | BS4 ionicons
--- | ---

**Description**

Create a ionicon.

**Usage**

ionicon(name)

**Arguments**

name | Name of icon. See https://ionic.io/ionicons/.
menuItemOutput

Note

Similar to the icon function from shiny.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Ionicons",
      body = dashboardBody(
        ionicon(name = "heart"),
        ionicon(name = "beer")
      ),
      server = function(input, output) {}
    ),
  )
}

menuItemOutput  Create a sidebar menu item output (client side)

Description

This is the UI-side function for creating a dynamic sidebar menu item.

Usage

menuItemOutput(outputId)

Arguments

outputId        Output variable name.

See Also

renderMenu for the corresponding server-side function and examples, and menuItem for the corresponding function for generating static sidebar menus.

Other menu outputs: dropdownMenuOutput(), menuOutput(), renderMenu(), sidebarMenuOutput()
menuOutput

Create a dynamic menu output for bs4Dash (client side)

Description

This can be used as a placeholder for dynamically-generated dropdownMenu, notificationItem, messageItem, taskItem sidebarMenu, or menuItem. If called directly, you must make sure to supply the correct type of tag. It is simpler to use the wrapper functions if present; for example, dropdownMenuOutput and sidebarMenuOutput.

Usage

```r
menuOutput(outputId, tag = shiny::tags$li)
```

Arguments

- `outputId`: Output variable name.
- `tag`: A tag function, like tags$li or tags$ul.

See Also

- `renderMenu` for the corresponding server side function and examples.
- Other menu outputs: dropdownMenuOutput(), menuItemOutput(), renderMenu(), sidebarMenuOutput()

popover

Create a Bootstrap 4 popover from the UI side

Description

This replaces the shinyBS popover feature that is not compatible with Bootstrap 4.

`addPopover` adds a popover to the given target.

`removePopover` destroys the current targeted popover.

Usage

```r
popover(tag, content, title, placement = c("top", "bottom", "left", "right"))
```

```r
dropDownMenuOutput(
  id = NULL,
  selector = NULL,
  options,
  session = shiny::getDefaultReactiveDomain()
)
```

```r
addPopover(id, session = shiny::getDefaultReactiveDomain())
```
popover

Arguments

- **tag**: Popover target.
- **content**: Popover content.
- **title**: Popover title.
- **placement**: Popover placement: "top", "bottom", "left" or "right".
- **id**: Popover target id.
- **selector**: jQuery selector. Allow more customization for the target (nested tags).
- **options**: List of options to pass to the popover. See [https://getbootstrap.com/docs/4.0/components/popovers/](https://getbootstrap.com/docs/4.0/components/popovers/).
- **session**: Shiny session object.

Note

**popover** does not automatically handles tooltip removal and must be seperately implemented. If the **dashboardHeader** help parameter is TRUE, all popovers may be enabled or disabled depending on the switch value, which may solve this problem. This allows to toggle popovers whenever required.

This replaces the shinyBS popover feature that is not compatible with Bootstrap 4.

Examples

```
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Popover UI",
      body = dashboardBody(
        popover(
          button("goButton", "Click me to see the popover!")
          title = "My popover",
          placement = "right",
          content = "Vivamus sagittis lacus vel augue laoreet rutrum faucibus."
        )
      )
    ),
    server = function(input, output) {}
  )
}
```

ui = dashboardPage(
    header = dashboardHeader(),
    sidebar = dashboardSidebar(),
    controlbar = dashboardControlbar(),
    footer = dashboardFooter(),
    title = "Popover server",
    body = dashboardBody(
        sliderInput("obs", "Number of observations:",
            min = 0, max = 1000, value = 500
        ),
        plotOutput("distPlot")
    )
),
server = function(input, output, session) {
    output$distPlot <- renderPlot({
        hist(rnorm(input$obs))
    })
    observeEvent(input$obs, {
        if (input$obs > 500) {
            addPopover(
                id = "distPlot",
                options = list(
                    content = "Vivamus sagittis lacus vel augue laoreet rutrum faucibus.",
                    title = "Server popover",
                    placement = "bottom",
                    trigger = "hover"
                )
            )
        } else {
            removePopover(id = "distPlot")
        }
    })
})
}

productList

AdminLTE3 product list container

Description

**productList** creates a container to display commercial items in an elegant container. Insert in a box.

**productListItem** creates a product item to insert in **productList**.

Usage

productList(...)

productListItem(..., image = NULL, title = NULL, subtitle = NULL, color = NULL)
Arguments

... product description.
image image url, if any.
title product name.
subtitle product price.
color price color. Valid color are listed below:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffc107.
  • danger: #dc3545.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

# Box with productList
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "Product List",
          status = "primary",
          productList(
            productListItem(
              image = "https://www.pngmart.com/files/1/Haier-TV-PNG.png",
              title = "Samsung TV",
              subtitle = "$1800",
              color = "warning",
              "This is an amazing TV, but I don't like TV!"
            ),
            productListItem(
              image = "https://upload.wikimedia.org/wikipedia/commons/7/77/IMac_Pro.svg",
              title = "Imac 27",
              subtitle = "$4999",
              color = "danger",
              "This is were I spend most of my time!"
            )
          )
        )
      )
    )
  )
}


renderbs4InfoBox

Bootstrap 4 info box

Description

A beautiful AdminLTE3 info box.

Usage

renderbs4InfoBox(expr, env = parent.frame(), quoted = FALSE)

bs4InfoBoxOutput(outputId, width = 4)

bs4InfoBox(
  title,
  value = NULL,
  subtitle = NULL,
  icon = shiny::icon("bar-chart"),
  color = NULL,
  width = 4,
  href = NULL,
  fill = FALSE,
  gradient = FALSE,
  elevation = NULL,
  iconElevation = NULL,
  tabName = NULL
)

infoBox(
  title,
  value = NULL,
  subtitle = NULL,
  icon = shiny::icon("bar-chart"),
  color = NULL,
  width = 4,
  href = NULL,
  fill = FALSE,
  gradient = FALSE,
  elevation = NULL,
renderbs4InfoBox

iconElevation = NULL,
tabName = NULL
)

infoBoxOutput(outputId, width = 4)

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

Arguments

expr An expression that returns a Shiny tag object, HTML(), or a list of such objects.

eval The parent environment for the reactive expression. By default, this is the calling environment, the same as when defining an ordinary non-reactive expression. If expr is a quosure and quoted is TRUE, then env is ignored.

quoted If it is TRUE, then the quote()ed value of expr will be used when expr is evaluated. If expr is a quosure and you would like to use its expression as a value for expr, then you must set quoted to TRUE.

outputId Output variable name.

width The width of the box, using the Bootstrap grid system. This is used for row-based layouts. The overall width of a region is 12, so the default width of 4 occupies 1/3 of that width. For column-based layouts, use NULL for the width; the width is set by the column that contains the box.

title Info box title.

value The value to display in the box. Usually a number or short text.

subtitle Any extra UI element.

icon An icon tag, created by icon.

color A color for the box. Valid colors are defined as follows:

- primary: #007bff.
- secondary: #6c757d.
- info: #17a2b8.
- success: #28a745.
- warning: #ffc107.
- danger: #dc3545.
- gray-dark: #343a40.
- gray: #adb5bd.
- white: #fff.
- indigo: #6610f2.
- lightblue: #3c8dbc.
- navy: #001f3f.
- purple: #605ca8.
- fuchsia: #f012be.
- pink: #e83e8c.
- maroon: #d81b60.
• orange: #ff851b.
• lime: #01ff70.
• teal: #39cccc.
• olive: #3d9970.

def
   An optional URL to link to.

fill
   If FALSE (the default), use a white background for the content, and the color
   argument for the background of the icon. If TRUE, use the color argument for
   the background of the content; the icon will use the same color with a slightly
   darkened background.

gradient
   Whether to use gradient style for background color. Default to FALSE.

elevation
   Box elevation.

iconElevation
   Icon elevation compared to the main content (relief). 3 by default.

tabName
   Optional: infoBox behaves like menuItem and may be used to navigate between
   multiple tabItem.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

Other cards: bs4CardLayout(), bs4SocialCard(), bs4TabCard(), bs4UserCard(), descriptionBlock(),
renderbs4ValueBox()

Examples

if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(
        sidebarMenu(
          menuItem(
            text = "Item 1",
            tabName = "tab1"
          ),
          menuItem(
            text = "Item 2",
            tabName = "tab2"
          )
        )
      ),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "test",
      body = dashboardBody(
    )
  )
)
renderbs4ValueBox

Description

This is the server-side function for creating a dynamic bs4ValueBox.

This is the UI-side function for creating a dynamic bs4ValueBox.

A beautiful AdminLTE3 value box.
Usage

renderbs4ValueBox(expr, env = parent.frame(), quoted = FALSE)

bs4ValueBoxOutput(outputId, width = 4)

bs4ValueBox(
  value,
  subtitle,
  icon = NULL,
  color = NULL,
  width = 3,
  href = NULL,
  footer = NULL,
  gradient = FALSE,
  elevation = NULL
)

valueBox(
  value,
  subtitle,
  icon = NULL,
  color = NULL,
  width = 3,
  href = NULL,
  footer = NULL,
  gradient = FALSE,
  elevation = NULL
)

valueBoxOutput(outputId, width = 4)

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

Arguments

expr
An expression that returns a Shiny tag object, HTML(), or a list of such objects.

event
The parent environment for the reactive expression. By default, this is the calling
environment, the same as when defining an ordinary non-reactive expression. If
expr is a quosure and quoted is TRUE, then env is ignored.

quoted
If it is TRUE, then the quoted value of expr will be used when expr is eval-
ulated. If expr is a quosure and you would like to use its expression as a value
for expr, then you must set quoted to TRUE.

outputId
Output variable name.

width
The width of the box, using the Bootstrap grid system. This is used for row-
based layouts. The overall width of a region is 12, so the default width of 4
occupies 1/3 of that width. For column-based layouts, use NULL for the width;
the width is set by the column that contains the box.
renderbs4ValueBox

value The value to display in the box. Usually a number or short text.
subtitle Subtitle text.
icon An icon tag, created by icon.
color The color of the item. This determines the item's background color. Valid colors are defined as follows:
- primary: #007bff.
- secondary: #6c757d.
- info: #17a2b8.
- success: #28a745.
- warning: #ffc107.
- danger: #dc3545.
- gray-dark: #343a40.
- gray: #adb5bd.
- white: #fff.
- indigo: #6610f2.
- lightblue: #3c8dbc.
- navy: #001f3f.
- purple: #605ca8.
- fuchsia: #f012be.
- pink: #e83e8c.
- maroon: #d81b60.
- orange: #ff851b.
- lime: #01ff70.
- teal: #39cccc.
- olive: #3d9970.
href An optional URL to link to in the footer. Should both 'footer' and this parameter be set, 'footer' will take precedence.
footer Optional html content for the footer of the box.
gradient Whether to use gradient style for background color. Default to FALSE.
elevation Value box elevation.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

bs4ValueBoxOutput for the corresponding UI-side function.
renderbs4ValueBox for the corresponding server-side function and examples.
Other cards: bs4CardLayout(), bs4SocialCard(), bs4TabCard(), bs4UserCard(), descriptionBlock(), renderbs4InfoBox()
Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shiny::shinyApp(
    ui = bs4DashPage(
      header = bs4DashNavbar(),
      sidebar = bs4DashSidebar(),
      controlbar = bs4DashControlbar(),
      footer = bs4DashFooter(),
      title = "test",
      body = bs4DashBody(
        fluidRow(
          bs4ValueBoxOutput("vbox"),
          bs4InfoBoxOutput("ibox")
        ),
      ),
    ),
    server = function(input, output) {
      output$vbox <- renderbs4ValueBox({
        bs4ValueBox(
          value = 150,
          subtitle = "New orders",
          status = "primary",
          icon = "shopping-cart",
          href = "#",
        )
      })

      output$ibox <- renderbs4InfoBox({
        bs4InfoBox(
          title = "Comments",
          gradientColor = "success",
          value = 41410,
          icon = "comments"
        )
      })
    }
  }
}
```

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "test",
      body = dashboardBody(
        fluidRow(
          bs4ValueBoxOutput("vbox"),
          bs4InfoBoxOutput("ibox")
        ),
      ),
    ),
    server = function(input, output) {
      output$vbox <- renderbs4ValueBox({
        bs4ValueBox(
          value = 150,
          subtitle = "New orders",
          status = "primary",
          icon = "shopping-cart",
          href = "#",
        )
      })

      output$ibox <- renderbs4InfoBox({
        bs4InfoBox(
          title = "Comments",
          gradientColor = "success",
          value = 41410,
          icon = "comments"
        )
      })
    }
  }
}
```
body = bs4DashBody(
    fluidRow(
        valueBox(
            value = 150,
            subtitle = "New orders",
            color = "primary",
            icon = icon("shopping-cart")
        ),
        valueBox(
            value = "53%",
            subtitle = "New orders",
            color = "indigo",
            icon = icon("cogs"),
            footer = div("Hello World")
        ),
        valueBox(
            value = "44",
            subtitle = "User Registrations",
            color = "teal",
            icon = icon("sliders")
        )
    ),
    server = function(input, output) {}
)

renderMenu

Create dynamic menu output (server side)

Description

Create dynamic menu output (server side)

Usage

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

Arguments

expr An expression that returns a Shiny tag object, HTML(), or a list of such objects.
env The parent environment for the reactive expression. By default, this is the calling environment, the same as when defining an ordinary non-reactive expression. If expr is a quosure and quoted is TRUE, then env is ignored.
quoted If it is TRUE, then the quoted value of expr will be used when expr is evaluated. If expr is a quosure and you would like to use its expression as a value for expr, then you must set quoted to TRUE.
outputArgs A list of arguments to be passed through to the implicit call to uiOutput() when renderUI is used in an interactive R Markdown document.
See Also

`menuItemOutput` for the corresponding client side function and examples.

Other menu outputs: `dropdownMenuOutput()`, `menuItemOutput()`, `menuOutput()`, `sidebarMenuOutput()`

Examples

```r
## Only run these examples in interactive R sessions

if (interactive()) {
  library(shiny)
  library(bs4Dash)
  messageData <- data.frame(
    from = c("Admininstrator", "New User", "Support"),
    message = c(
      "Sales are steady this month.",
      "How do I register?",
      "The new server is ready."
    ),
    stringsAsFactors = FALSE
  )

  # ========== Dynamic dropdownMenu ==========
  ui <- dashboardPage(
    dashboardHeader(
      title = "Dynamic menus",
      dropdownMenuOutput("messageMenu")
    ),
    dashboardSidebar(),
    dashboardBody(
      fluidRow(
        box(
          title = "Controls",
          sliderInput("slider", "Number of observations:" , 1, 100, 50)
        )
      )
    )
  )
  server <- function(input, output) {
    output$messageMenu <- renderMenu({
      # Code to generate each of the messageItems here, in a list. messageData
      # is a data frame with two columns, 'from' and 'message'.
      # Also add on slider value to the message content, so that messages update.
      msgs <- apply(messageData, 1, function(row) {
        messageItem(
          from = row["from"],
          message = paste(row["message"], input$slider)
        )
      })
      dropdownMenu(type = "messages", .list = msgs)
    })
  }
}
```
Create a sidebar menu output (client side)

Description

This is the UI-side function for creating a dynamic sidebar menu.

Usage

`sidebarMenuOutput(outputId)`

Arguments

`outputId` Output variable name.

See Also

`renderMenu` for the corresponding server-side function and examples, and `sidebarMenu` for the corresponding function for generating static sidebar menus.

Other menu outputs: `dropdownMenuOutput()`, `menuItemOutput()`, `menuOutput()`, `renderMenu()`
**skinSelector**  
*AdminLTE3 skin selector*

**Description**

This creates a skin selector element.

**Usage**

skinSelector()

**Examples**

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(
        sidebarMenu(
          menuItem(
            text = "Item 1"
          ),
          menuItem(
            text = "Item 2"
          )
        ),
        body = dashboardBody(),
        controlbar = dashboardControlbar(skinSelector(), pinned = TRUE),
        title = "Skin Selector"
      ),
      server = function(input, output) { }
    )
  )
}
```

**tabsetPanel**  
*Create a tabsetPanel*

**Description**

Imported by `bs4TabCard` but can be used alone. This is a modified shiny::tabsetPanel, to handle bootstrap 4. This function will be upgraded starting from shiny 1.7.0 (support Bootstrap 4 tabs).
tabsetPanel

Usage

```
tabsetPanel(
  ..., id = NULL,
  selected = NULL,
  type = c("tabs", "pills", "hidden"),
  vertical = FALSE,
  side = "left",
  .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
- `vertical` Whether to displays tabs vertically. Default to FALSE.
- `side` Tabs side: "left" or "right".
- `.list` In case of programmatically generated items. See example.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```
if(interactive()){
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Bootstrap 4 tabsetPanel",
      body = dashboardBody(
        # manually inserted panels
        tabsetPanel(
          id = "tabcard",
          tabPanel(
            title = "Tab 1",
          ),
          tabPanel(
            title = "Tab 2",
          ),
          tabPanel(
            title = "Tab 3",
          ),
        ),
      ),
    ),
  )
}
```
"Content 1"
),
tabPanel(
    title = "Tab 2",
    "Content 2"
),
tabPanel(
    title = "Tab 3",
    "Content 3"
),
br(), br(),
# programmatically inserted panels
tabsetPanel(
    id = "tabset",
    .list = lapply(1:3, function(i) {
        tabPanel(
            title = paste0("Tab", i),
            active = FALSE,
            paste("Content", i)
        )
    })
),
),
server = function(input, output) {}
)

# update tabsetPanel
shinyApp(
    ui = dashboardPage(
        title = "updateTabsetPanel",
        header = dashboardHeader(),
        body = dashboardBody(
            tabsetPanel(
                id = "tabset1",
                selected = "Tab 2",
                tabPanel(
                    title = "Tab 1",
                    numericInput("val", "Value:", 10, min = 1, max = 100),
                    verbatimTextOutput("value")
                ),
                tabPanel(
                    title = "Tab 2",
                    "Content 2"
                ),
                tabPanel(
                    title = "Tab 3",
                    checkboxGroupInput(
                        inline = TRUE,
                        "variable", "Variables to show:",
                        c("Cylinders" = "cyl",}
tabsetPanel

   "Transmission" = "am",
   "Gears" = "gear")
),
tableOutput("data")
)
)
uoOutput("tabSetPanel2")
),
sidebar = dashboardSidebar(
   skin = "light",
sliderInput(
      inputId = "controller",
      label = "Update the first tabset",
      min = 1,
      max = 3,
      value = 2
   ),
br(),
sliderInput(
      inputId = "controller2",
      label = "Update the second tabset",
      min = 1,
      max = 3,
      value = 3
   ),
controlbar = dashboardControlbar(collapsed = FALSE),
footer = dashboardFooter()
),
server = function(input, output, session) {

   output$tabSetPanel2 <- renderUI({
      tabsetPanel(
         id = "tabset2",
         tabPanel(
            title = "Tab 1",
            p("Tab 1")
         ),
         tabPanel(
            title = "Tab 2",
            p("Tab 2")
         ),
         tabPanel(
            title = "Tab 3",
            p("Tab 3")
         )
      )
   })

   # update tabset1
   observeEvent(input$controller, {
      updateTabsetPanel(
         session,
         "Transmission" = input$controller,
         "Gears" = input$controller
      )
   })
toast

Create an adminLTE toast

Description

Builtin AdminLTE3 toasts

Usage

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

- **title**: Toast title.
- **body**: Body content.
- **subtitle**: Toast subtitle.
- **options**: Toasts options: a list. See [https://adminlte.io/docs/3.0/javascript/toasts.html](https://adminlte.io/docs/3.0/javascript/toasts.html).
- **session**: Shiny session object.

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        actionButton("sendToast", "Send Toast")
      ),
      controlbar = dashboardControlbar(),
      title = "Toasts"
    ),
    server = function(input, output) {
      observeEvent(input$sendToast, {
        toast(
          title = "My Toast",
          body = h4("I am a toast!")
        ,
          options = list(
            autohide = TRUE,
            icon = "fas fa-home",
            close = FALSE
          )
        )
      })
    }
  )
}
```

---

**tooltip**

*Create a Bootstrap 4 Tooltip from the UI side*

**Description**

This replaces the shinyBS tooltip feature that is not compatible with Bootstrap 4.

- `addTooltip` adds a tooltip to the given target.
- `removeTooltip` destroys the current targeted tooltip.
Usage

```r
tooltip(tag, title, placement = c("top", "bottom", "left", "right"))
```

```r
addTooltip(
  id = NULL,
  selector = NULL,
  options,
  session = shiny::getDefaultReactiveDomain()
)
```

```r
removeTooltip(id, session = shiny::getDefaultReactiveDomain())
```

Arguments

- **tag**: Tooltip target.
- **title**: Tooltip title.
- **placement**: Tooltip placement: "top", "bottom", "left" or "right".
- **id**: Tooltip target id.
- **selector**: jQuery selector. Allow more customization for the target (nested tags).
- **options**: List of options to pass to the tooltip. See https://getbootstrap.com/docs/4.0/components/tooltips/.
- **session**: Shiny session object.

Note

`tooltip` does not automatically handles tooltip removal and must be seperately implemented. If the `dashboardHeader` help parameter is TRUE, all tooltips may be enabled or disabled depending on the switch value, which may solve this problem. This allows to toggle tooltips whenever required. This replaces the shinyBS tooltip feature that is not compatible with Bootstrap 4

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      controlbar = dashboardControlbar(),
      footer = dashboardFooter(),
      title = "Tooltip UI",
      body = dashboardBody(
        tooltip(
          actionButton("goButton", "Hover to see the tooltip"),
          title = "My tooltip",
          placement = "top"
        )
      )
    )
  )
}
```
if (interactive()) {
    library(shiny)
    library(bs4Dash)

    shinyApp(
        ui = dashboardPage(
            header = dashboardHeader(),
            sidebar = dashboardSidebar(),
            controlbar = dashboardControlbar(),
            footer = dashboardFooter(),
            title = "Tooltip server",
            body = dashboardBody(
                sliderInput("obs", "Number of observations:",
                    min = 0, max = 1000, value = 500
                ),
                plotOutput("distPlot")
            )
        ),
        server = function(input, output, session) {
            output$distPlot <- renderPlot({
                hist(rnorm(input$obs))
            })

            observeEvent(input$obs, {
                if (input$obs > 500) {
                    addTooltip(
                        id = "distPlot",
                        options = list(
                            title = "Server tooltip",
                            placement = "bottom"
                        )
                    )
                } else {
                    removeTooltip(id = "distPlot")
                }
            })
        }
    )
}
useAutoColor

Description

This piece of code is necessary so that plots get the good background color, automatically. It requires the use of the thematic package and shiny dev.

Usage

useAutoColor(input, output, session = shiny::getDefaultReactiveDomain())

Arguments

- **input**: Shiny input object.
- **output**: Shiny output object.
- **session**: Shiny session object.

Value

An observer telling Shiny to update the current theme. It has to be inserted at the top of the main server function.

Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)
  library(thematic)

  thematic_shiny()
  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(
        title = bs4DashBrand(
          title = "My dashboard",
          color = "primary",
          href = "https://adminlte.io/themes/v3",
          image = "https://adminlte.io/themes/v3/dist/img/AdminLTELogo.png"
        ),
      ),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        sliderInput("obs", "Number of observations:",
          min = 0, max = 1000, value = 500
        ),
        plotOutput("distPlot")
      ),
      controlbar = dashboardControlbar(),
      title = "DashboardPage"
    ),
    server = function(input, output, session) {
      useAutoColor()
      output$distPlot <- renderPlot({
        hist(rnorm(input$obs))
      })
    }
  )
}
```
userList  

AdminLTE3 user list container

Description

userList creates a user list container to be inserted in a box.
userListItem creates a user list item.

Usage

userList(...)

userListItem(image, title, subtitle = NULL)

Arguments

... slot for userListItem.
image image url or path.
title Item title.
subtitle Item subtitle.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "User List example",
          status = "success",
          userList(
            userListItem(
              image = "https://adminlte.io/themes/v3/dist/img/user1-128x128.jpg",
              title = "Shiny",
            )
          )
        )
      )
    )
  )
}
userMessages

AdminLTE3 user message container

Description

userMessages creates a user message container. Maybe inserted in a box.
userMessage creates a user message html element.
updateUserMessages allows to interact with a userMessages container, such as sending, removing or editing messages.

Usage

userMessages(..., id = NULL, status, width = 4, height = NULL)

userMessage(
  ..., author = NULL, date = NULL, image = NULL, type = c("sent", "received")
)

updateUserMessages(
  id, action = c("add", "remove", "update"), index = NULL,
)
userMessages

content = NULL,
session = shiny::getDefaultReactiveDomain()
)

Arguments

... Message text.
id userMessages to target.
status Messages status. Valid colors are defined as follows:
  • primary: #007bff.
  • secondary: #6c757d.
  • info: #17a2b8.
  • success: #28a745.
  • warning: #ffc107.
  • danger: #dc3545.
  • gray-dark: #343a40.
  • gray: #adb5bd.
  • white: #fff.
  • indigo: #6610f2.
  • lightblue: #3c8dbc.
  • navy: #001f3f.
  • purple: #605ca8.
  • fuchsia: #f012be.
  • pink: #e83e8c.
  • maroon: #d81b60.
  • orange: #ff851b.
  • lime: #01ff70.
  • teal: #39ccc.
  • olive: #3d9970.
width Container width: between 1 and 12.
height Container height.
author Message author.
date Message date.
image Message author image path or url.
type Message type: c("sent","received").
action Action to perform: add, remove or update.
index Index of item to update or remove.
content New message content in a list. For actions like add and update only! See example.
session Shiny session object.
Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if (interactive()) {
library(shiny)
library(bs4Dash)

shinyApp(
  ui = dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
      box(
        title = "Box with messages",
        solidHeader = TRUE,
        status = "warning",
        userMessages(
          width = 12,
          status = "teal",
          userMessage(
            author = "Alexander Pierce",
            date = "20 Jan 2:00 pm",
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user1-128x128.jpg",
            type = "sent",
            "Is this template really for free? That's unbelievable!"
          ),
          userMessage(
            author = "Sarah Bullock",
            date = "23 Jan 2:05 pm",
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user3-128x128.jpg",
            type = "received",
            "You better believe it!"
          )
        )
      ),
      userMessages(
        width = 6,
        status = "danger",
        userMessage(
          author = "Alexander Pierce",
          date = "20 Jan 2:00 pm",
          image = "https://adminlte.io/themes/AdminLTE/dist/img/user1-128x128.jpg",
          type = "received",
          "Is this template really for free? That's unbelievable!"
        ),
        userMessage(
          author = "Sarah Bullock",
          date = "23 Jan 2:05 pm",
          image = "https://adminlte.io/themes/AdminLTE/dist/img/user3-128x128.jpg",
          type = "sent",
```
"You better believe it!"

dealer_message,

server = function(input, output) {

if (interactive()) {

library(shiny)
library(bs4Dash)

shinyApp{

ui = dashboardPage(
  dashboardHeader(),
  dashboardSidebar(),
  dashboardBody(
    fluidRow(
      actionButton("remove", "Remove message"),
      actionButton("add", "Add message"),
      actionButton("update", "Update message")
    ),
    numericInput("index", "Message index:", 1, min = 1, max = 3),
    br(),
    br(),
    userMessages(
      width = 6,
      status = "danger",
      id = "message",
      userMessage(
        author = "Alexander Pierce",
        date = "20 Jan 2:00 pm",
        image = "https://adminlte.io/themes/AdminLTE/dist/img/user1-128x128.jpg",
        type = "received",
        "Is this template really for free? That's unbelievable!"
      ),
      userMessage(
        author = "Sarah Bullock",
        date = "23 Jan 2:05 pm",
        image = "https://adminlte.io/themes/AdminLTE/dist/img/user3-128x128.jpg",
        type = "sent",
        "You better believe it!"
      )
    )
  ),
  title = "user Message"
),
server = function(input, output, session) {

observeEvent(input$remove, {
  updateUserMessages("message", action = "remove", index = input$index)
})

}
observeEvent(input$add, {
  updateUserMessages(
    "message",
    action = "add",
    content = list(
      author = "David",
      date = "Now",
      image = "https://i.pinimg.com/originals/f1/15/df/f115dfc9c0b063597b1221d015996b39.jpg",
      type = "received",
      text = tagList(
        sliderInput(
          "obs",
          "Number of observations:",
          min = 0,
          max = 1000,
          value = 500
        ),
        plotOutput("distPlot")
      )
    )
  )
})

output$distPlot <- renderPlot({
  hist(rnorm(input$obs))
})

observeEvent(input$update, {
  updateUserMessages(
    "message",
    action = "update",
    index = input$index,
    content = list(
      text = tagList(
        appButton(
          inputId = "reload",
          label = "Click me!",
          icon = icon("sync"),
          dashboardBadge(1, color = "primary")
        )
      )
    )
  )
})

observeEvent(input$reload, {
  showNotification("Yeah!", duration = 1, type = "default")
})
userPost

AdminLTE3 user post

Description

Creates a user post. This content may be inserted in a box.
userPostTagItems creates a container to host userPostTagItem.
userPostTagItem creates a user post tool item

Usage

userPost(

..., id = NULL, image, author, description = NULL, collapsible = TRUE, collapsed = FALSE

)

userPostTagItems(...)

userPostTagItem(...)

Arguments

... Tool content such as label, button, ...
id Unique id of the post.
image Profile image, if any.
author Post author.
description Post description.
collapsible If TRUE, display a button in the upper right that allows the user to collapse the comment.
collapsed Whether the comment is collapsed when the application starts, FALSE by default.

Author(s)

David Granjon, <dgranjon@ymail.com>

See Also

Other boxWidgets: attachmentBlock(), bs4CardLabel(), bs4CardSidebar(), bs4Carousel(), bs4SocialCard(), bs4Timeline(), cardDropdown(), cardProfile(), descriptionBlock()
Examples

```r
if (interactive()) {
  library(shiny)
  library(bs4Dash)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "Box with user comment",
          status = "primary",
          userPost(
            id = 1,
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user1-128x128.jpg",
            author = "Jonathan Burke Jr.",
            description = "Shared publicly - 7:30 PM today",
            "Lorem ipsum represents a long-held tradition for designers,
            typographers and the like. Some people hate it and argue for
            its demise, but others ignore the hate as they create awesome
            tools to help create filler text for everyone from bacon
            lovers to Charlie Sheen fans.",
            collapsible = FALSE,
            userPostTagItems(
              userPostTagItem(dashboardBadge("item 1", color = "info")),
              userPostTagItem(dashboardBadge("item 2", color = "danger"), side = "right")
            )
          ),
          userPost(
            id = 2,
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user6-128x128.jpg",
            author = "Adam Jones",
            userPostMedia(image = "https://adminlte.io/themes/AdminLTE/dist/img/photo2.png"),
            userPostTagItems(
              userPostTagItem(dashboardBadge("item 1", color = "success")),
              userPostTagItem(dashboardBadge("item 2", color = "danger"), side = "right")
            )
          ),
          title = "userPost"
        ),
        server = function(input, output) { }
    )
  )
}
```
**userPostMedia**

**Description**

`userPostMedia` creates a container to include an image in `userPost`.

**Usage**

```
userPostMedia(image, height = NULL, width = NULL)
```

**Arguments**

- `image`: Image path or url ...
- `height`: Media height in pixels.
- `width`: Media width in pixels.

**Author(s)**

David Granjon, <dgranjon@ymail.com>
Index

* **boxWidgets**
  attachmentBlock, 7
  bs4CardLabel, 15
  bs4CardSidebar, 17
  bs4Carousel, 20
  bs4SocialCard, 56
  bs4Timeline, 67
cardDropdown, 77
cardProfile, 78
descriptionBlock, 83
userPost, 127

* **cards**
  bs4CardLayout, 15
  bs4SocialCard, 56
  bs4TabCard, 62
  bs4UserCard, 70
descriptionBlock, 83
renderbs4InfoBox, 102
renderbs4ValueBox, 105

* **input elements**
  actionButton, 3

* **menu items**
  bs4DropdownMenu, 40

* **menu outputs**
  dropdownMenuOutput, 94
  menuItemOutput, 97
  menuOutput, 98
  renderMenu, 109
  sidebarMenuOutput, 111

* **user outputs**
  bs4UserMenu, 75

accordion, 9
accordion (bs4Accordion), 9
accordionItem, 9, 10
accordionItem (bs4Accordion), 9
actionButton, 3, 12, 43
addPopover, 98
addPopover (popover), 98
addTooltip (popover), 117
addTooltip (tooltip), 117
appButton, 6
attachmentBlock, 7, 7, 15, 19, 20, 58, 69, 78, 79, 88, 127
blockQuote (bs4Quote), 53
box, 7, 18, 60, 67, 78, 83, 100, 121, 122, 127
box (descriptionBlock), 83
boxComment (bs4SocialCard), 56
boxDropdown, 58, 65, 73, 77, 88
boxDropdown (cardDropdown), 77
boxDropdownItem (cardDropdown), 77
boxLabel, 58, 65, 73, 88
boxLabel (bs4CardLabel), 15
boxLayout (bs4CardLayout), 15
boxPad, 83, 86
boxPad (descriptionBlock), 83
boxProfile, 78
boxProfile (cardProfile), 78
boxProfileItem, 79
boxProfileItem (cardProfile), 78
boxSidebar, 58, 65, 73, 88
boxSidebar (bs4CardSidebar), 17
bs4Accordion, 9
bs4AccordionItem (bs4Accordion), 9
bs4Badge, 12
bs4Callout, 13
bs4Card (descriptionBlock), 83
bs4CardLabel, 8, 15, 15, 19, 20, 58, 69, 78, 79, 88, 127
bs4CardLayout, 15, 15, 58, 65, 74, 88, 104, 107
bs4CardSidebar, 8, 15, 17, 20, 58, 69, 78, 79, 88, 127
bs4Carousel, 8, 15, 19, 20, 58, 69, 78, 79, 88, 127
bs4CarouselItem (bs4Carousel), 20
bs4DashBody, 21, 29
bs4DashBrand, 22, 22
bs4DashControlbar, 23, 29
INDEX

updatebs4TabItems (bs4DashSidebar), 31
updateCard (descriptionBlock), 83
updateCardSidebar (bs4DashSidebar), 17
updateControlbar (bs4DashControlbar), 23
updateControlbarMenu
  (bs4DashControlbar), 23
updateSidebar, 31
updateSidebar (bs4DashSidebar), 31
updateTabItems, 31
updateTabItems (bs4DashSidebar), 31
updateTabsetPanel, 31
updateUserMessages, 122
useAutoColor, 119
userBlock, 57
userBlock (bs4SocialCard), 56
userBox, 70
userBox (bs4UserCard), 70
userDescription, 70
userDescription (bs4UserCard), 70
userList, 121, 121
userListItem, 121
userListItem (userList), 121
userMessage, 122
userMessage (userMessages), 122
userMessages, 122, 122, 123
userOutput, 77
userOutput (bs4UserMenu), 75
userPost, 8, 15, 19, 20, 58, 69, 78, 79, 88, 127, 129
userPostMedia, 128, 129
userPostTagItem, 127
userPostTagItem (userPost), 127
userPostTagItems, 127
userPostTagItems (userPost), 127
validateCssUnit(), 6
validColors, 64, 72, 87
valueBox (renderbs4ValueBox), 105
valueBoxOutput (renderbs4ValueBox), 105
waiterShowOnLoad, 30