Package ‘shinydashboardPlus’

September 15, 2021

Type Package

Title Add More 'AdminLTE2' Components to 'shinydashboard'

Version 2.0.3

Maintainer David Granjon <dgranjon@ymail.com>

Description Extend 'shinydashboard' with 'AdminLTE2' components. 'AdminLTE2' is a free 'Bootstrap 3' dashboard template available at <https://adminlte.io>. Customize boxes, add timelines and a lot more.

License GPL (>= 2) | file LICENSE

Imports shiny,
      htmltools,
      shinydashboard,
      fresh,
      waiter (>= 0.2.3),
      lifecycle

Suggests styler (>= 1.2.0),
       shinyAce,
       shinyWidgets,
       shinyEffects,
       shinyjqui,
       knitr,
       rmarkdown,
       jsonlite


BugReports https://github.com/RinteRface/shinydashboardPlus/issues

Encoding UTF-8

RoxygenNote 7.1.2

VignetteBuilder knitr

RdMacros lifecycle

\R topics documented:

accordion .......................................................... 2
appButton ................................................................ 5
attachmentBlock ...................................................... 6
accordian

Description

Create an accordion container. Accordions are part of collapsible elements.

`accordionItem` creates an accordion item to put inside an `accordion` container.

`updateAccordion` toggles an `accordion` on the client.
Usage

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

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

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

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: #3c8dbc
• success: #00a65a
• info: #00c0ef
• warning: #f39c12
• danger: #f56954
• navy: #001F3F
• teal: #39CCCC
• purple: #605ca8
• orange: #ff851b
• maroon: #D81B60
• black: #111111

Only primary, success, info, warning and danger are compatible with solidHeader!

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(shinydashboard)
library(shinydashboardPlus)

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 = "warning",  
    collapsed = FALSE,  
    "This is some text!"  
  )  
),
accordion(  
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!"  
  )  
)

# Update accordion
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  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,
appButton

"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")
  })
}
}

---

appButton AdminLTE2 special large button

Description

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

Usage

appButton(..., inputId, label, icon = NULL, width = 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().

Author(s)

David Granjon, <dgranjon@ymail.com>
Examples

```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

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

attachmentBlock

AdminLTE2 attachment container

Description

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

Usage

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

Arguments

- `...` any element.
- `image` url or path to the image.
- `title` attachment title.
- `href` external link.

Author(s)

David Granjon, <dgranjon@ymail.com>
blockQuote

Examples

```r
# Box with attachmentBlock
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "Attachment example",
          attachmentBlock(
            image = "https://adminlte.io/themes/AdminLTE/dist/img/photo1.png",
            title = "Test",
            href = "https://google.com",
            "This is the content"
          )
        ),
        title = "AttachmentBlock"
      ),
      server = function(input, output) {
      }
    )
  )
```

blockQuote

AdminLTE2 block quote

Description

If you want to quote text

Usage

```r
blockQuote(..., side = "left")
```

Arguments

- `...` any element.
- `side` blockquote orientation. "left" by default, can be set to "right".

Author(s)

David Granjon, <dgranjon@ymail.com>
Examples

```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "BlockQuote example",
          blockQuote("I quote some text here!"),
          blockQuote("I quote some text here!", side = "right")
        ),
        title = "blockQuote"
      ),
      server = function(input, output) { }
    )
  )
}
```

box  
Create a box for the main body of a dashboard

Description

box can be used to hold content in the main body of a dashboard.
updateBox is used to toggle, close or restore a box on the client.
boxDropdown is used in the dropdown parameter of box.
boxDropdownItem goes in boxDropdown.
dropdownDivider goes in boxDropdown but also in any dropdown menu container.
boxPad creates a vertical container for descriptionBlock. It has to be included in a box.
descriptionBlock creates a description block, perfect for writing statistics to insert in a box.

Usage

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

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

boxDropdown(..., icon = shiny::icon("wrench"))

boxDropdownItem(..., id = NULL, href = NULL, icon = NULL)
dropdownDivider()

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

descriptionBlock(
  number = NULL,
  numberColor = NULL,
  numberIcon = NULL,
  header = NULL,
  text = NULL,
  rightBorder = TRUE,
  marginBottom = FALSE
)

Arguments

... any element such as descriptionBlock.
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: #3c8dbc
  • success: #00a65a
  • info: #00c0ef
  • warning: #f39c12
  • danger: #f56954
  • navy: #001f3f
  • teal: #39cccc
  • purple: #605ca8
• orange: #ff851b
• maroon: #D81B60
• black: #111111

Only primary, success, info, warning and danger are compatible with solid-
Header!

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:
• light-blue (primary status): #3c8dbc.
• red (danger status): #dd4b39.
• green (success status): #00a65a.
• aqua (info status): #00c0ef.
• yellow (warning status): #f39c12.
• blue: #0073b7.
• navy: #001F3F.
• teal: #39CCCC.
• olive: #3D9970.
• lime: #01FF70.
• orange: #FF851B.
• fuchsia: #F012BE.
• purple: #605ca8.
• maroon: #D81B60.
• black: #111.
• gray: #d2d6de.

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 value Box
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.

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.

icon Optional icon. 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".

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 If passed, the item will behave like an action button.

action Action to trigger: either collapse, remove, restore or 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)`. If the box had a background/status (any item that may be NULL), you must explicitly pass background = NULL, if you want to remove the background value.

session

Shiny session object.

target

Target url or page.

color

background color: see here for a list of valid colors https://adminlte.io/themes/AdminLTE/pages/UI/general.html. See below:

- light-blue (primary status): #3c8dbc.
- red (danger status): #dd4b39.
- green (success status): #00a65a.
- aqua (info status): #00c0ef.
- yellow (warning status): #f39c12.
- blue: #0073b7.
- navy: #001F3F.
- teal: #39CCCC.
- olive: #3D9970.
- lime: #01FF70.
- orange: #FF851B.
- fuchsia: #F012BE.
- purple: #605ca8.
- maroon: #D81B60.
- black: #111.
- gray: #d2d6de.

style

custom CSS, if any.

number

any number.

numberColor

number color: see here for a list of valid colors https://adminlte.io/themes/AdminLTE/pages/UI/general.html. See below:

- light-blue (primary status): #3c8dbc.
- red (danger status): #dd4b39.
- green (success status): #00a65a.
- aqua (info status): #00c0ef.
- yellow (warning status): #f39c12.
- blue: #0073b7.
- navy: #001F3F.
- teal: #39CCCC.
- olive: #3D9970.
- lime: #01FF70.
- orange: #FF851B.
- fuchsia: #F012BE.
- purple: #605ca8.
- maroon: #D81B60.
- black: #111.
- gray: #d2d6de.
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.

Examples

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

  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",
            style = "circle"
          ),
          dropdownMenu = boxDropdown(
            boxDropdownItem("Link to google", href = "http://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")
          )
        ),
        plotOutput("distPlot")
      )
    ),
    server = function(input, output) {
      
    }
  )
}
output$distPlot <- renderPlot({
  hist(rnorm(input$obs))
})

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

  ui <- dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
      tags$style("body { background-color: ghostwhite}\"),
      fluidRow(
        actionButton("toggle_box", "Toggle Box"),
        actionButton("remove_box", "Remove Box", class = "bg-danger"),
        actionButton("restore_box", "Restore Box", class = "bg-success"),
      ),
      actionButton("update_box", "Update Box", class = "bg-info"),
      actionButton("update_box2", "Update Box 2", class = "bg-info"),
      br(),
      br(),
      box(
        title = textOutput("box_state"),
        id = "mybox",
        status = "danger",
        background = "maroon",
        gradient = TRUE,
        collapsible = TRUE,
        closable = TRUE,
        plotOutput("plot")
      )
    )
  )

  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", dashboardLabel(!, status = "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)

# Box with dropdown items and input
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)
  shinyApp(
    ui = dashboardPage(
      ...)
dashboardHeader(),
dashboardSidebar(),
dashboardBody(
  box(
    title = "Closable Box with dropdown",
    closable = TRUE,
    width = 12,
    status = "warning",
    solidHeader = FALSE,
    collapsible = TRUE,
    dropdownMenu = boxDropdown(
      boxDropdownItem("Click me", id = "dropdownItem", icon = icon("heart")),
      boxDropdownItem("item 2", href = "https://www.google.com"),
      dropdownDivider(),
      boxDropdownItem("item 3", icon = icon("th"))
    ),
    "My box"
  )
),
server = function(input, output) {
  observeEvent(input$dropdownItem, {
    showNotification("Hello", duration = 1, type = "message")
  })
}
)

# Box with boxPad container + descriptionBlock
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(title = "Box with right pad",
            status = "warning",
            fluidRow(
              column(width = 6),
              column(
                width = 6,
                boxPad(
                  color = "green",
                  descriptionBlock(
                    header = "8390",
                    text = "VISITS",
                    rightBorder = FALSE,
                    marginBottom = TRUE
                  ),
                  descriptionBlock(
                    header = "30%",
                    text = "REFERRALS",
                    rightBorder = FALSE,
                    marginBottom = TRUE
                  )
                )
              )
            )
        )
      )
    )
  )
}
```r
if (interactive()) {
library(shiny)
library(shinydashboard)
library(shinydashboardPlus)

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 = "green",
              numberIcon = icon("caret-up"),
              header = "$35,210.43",
              text = "TOTAL REVENUE",
              rightBorder = TRUE,
              marginBottom = TRUE
            )
          ),
          column(
            width = 6,
            descriptionBlock(
              number = "18%",
              numberColor = "red",
              numberIcon = icon("caret-down"),
              header = "1200",
```
**boxLabel**

Create a label for box

**Description**

boxLabel is inserted in the label slot of box.

**Usage**

boxLabel(text, status, style = "default")

**Arguments**

- **text**
  - Label text. In practice only few letters or a number.

- **status**
  - label color status. See https://adminlte.io/themes/AdminLTE/pages/UI/general.html. Valid statuses are defined as follows:
    - primary: #3c8dbc
    - success: #00a65a
    - info: #00c0ef
    - warning: #f39c12
    - danger: #f56954

- **style**
  - label border style: "default" (rounded angles), "circle" or "square".

**boxProfile**

AdminLTE2 box profile

**Description**

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

boxProfileItem is an sub-element of a boxProfile.
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>

Examples

# Box with boxProfile
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "Box with profile",
          status = "primary",
          boxProfile(
            image = "https://adminlte.io/themes/AdminLTE/dist/img/user4-128x128.jpg",
            title = "Nina Mcintire",
            subtitle = "Software Engineer",
            bordered = TRUE,
            boxProfileItem(
              title = "Followers",
              description = 1322
            ),
            boxProfileItem(
              title = "Following",
              description = 543
            ),
            boxProfileItem(
              title = "Friends",
              description = 13287
            )
          )
        )
      )
    ),
  )
)
boxSidebar

Create a sidebar for a box

Description

boxSidebar is inserted in the sidebar slot of box.
updateBoxSidebar toggle a boxSidebar on the client.

Usage

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

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

Arguments

...  Sidebar content.
id  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. Expect a HEX code.
startOpen  Whether the sidebar is open at start. FALSE by default.
icon  Sidebar icon. Expect icon.
session  Shiny session object.

Examples

# Toggle a box sidebar
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),

```
carousel

AdminLTE2 carousel container

description

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

usage

carousel(..., id, indicators = TRUE, width = 6, .list = NULL)
carouselItem(..., caption = NULL, active = FALSE)

arguments

... Element such as images, iframe, ...
id Carousel id. Must be unique.
indicators Whether to display left and right indicators.
width Carousel width. 6 by default.
.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.
**dashboardBadge**

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

**Examples**
```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        carousel(
          id = "mycarousel",
          carouselItem(
            caption = "Item 1",
            tags$img(src = "https://placehold.it/900x500/3c8dbc/ffffff&text=I+Love+Bootstrap") ),
          carouselItem(
            caption = "Item 2",
            tags$img(src = "https://placehold.it/900x500/39CCCC/ffffff&text=I+Love+Bootstrap")
          )
        ),
        title = "Carousel"
      ),
      server = function(input, output) { }
  )
}
```

**dashboardBadge**  
**AdminLTE2 badge**

**Description**
Create a badge. It may be inserted in any element like inside a `actionButton` or a `dashboardSidebar`.

**Usage**
`dashboardBadge(..., color)`

**Arguments**
- `...` Any html text element.
- `color` label color. See below:
  - light-blue (primary status): #3c8dbc.
  - red (danger status): #dd4b39.
  - green (success status): #00a65a.
  - aqua (info status): #00c0ef.
• yellow (warning status): #f39c12.
• blue: #0073b7.
• navy: #001F3F.
• teal: #39CCCC.
• olive: #3D9970.
• lime: #01FF70.
• orange: #FF851B.
• fuchsia: #F012BE.
• purple: #605ca8.
• maroon: #D81B60.
• black: #111.
• gray: #d2d6de.

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

Examples
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

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

dashboardControlbar create a right sidebar container.
updateControlbar allows to toggle a dashboardControlbar.
controlbarMenu is a tabset panel for the dashboardControlbar.
controlbarItem is a tabPanel for the controlbarMenu.

**Usage**

dashboardControlbar(
  ...
  id = NULL,
  disable = FALSE,
  width = 230,
  collapsed = TRUE,
  overlay = TRUE,
  skin = "dark",
  .list = NULL
)

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

controlbarMenu(..., id = NULL, selected = NULL)

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

updateControlbarMenu(
  id,
  selected = NULL,
  session = shiny::getDefaultReactiveDomain()
)

**Arguments**

... slot for controlbarMenu. Not compatible with .items.

id Controlbar id.

disable If TRUE, the sidebar will be disabled.

width Sidebar width in pixels. Numeric value expected. 230 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 background color: "dark" or "light".

.list Pass element here if you do not want to embed them in panels. Not compatible with ...

session Shiny session object.

selected Item to select.

title Display title for tab
dashboardControlbar

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().

Note

Until a maximum of 5 controlbarItem! AdminLTE 2 does not support more panels.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

# Controlbar example
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)
  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(),
      controlbar = dashboardControlbar(
        skin = "dark",
        controlbarMenu(
          id = "menu",
          controlbarItem("Tab 1",
            "Welcome to tab 1"
          ),
          controlbarItem("Tab 2",
            "Welcome to tab 2"
          )
        )
      ),
      title = "Right Sidebar"
    ),
    server = function(input, output) { }
  )
}

# Toggle the dashboard controlbar
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)
  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        controlbar = dashboardControlbar(
          skin = "dark",
          controlbarMenu(
            id = "menu",
            controlbarItem("Tab 1",
              "Welcome to tab 1"
            ),
            controlbarItem("Tab 2",
              "Welcome to tab 2"
            )
          )
        ),
        title = "Right Sidebar"
      )
    ),
    server = function(input, output) { }
  )
}
dashboardControlbar

```
actionButton(inputId = "controlbarToggle", label = "Toggle Controlbar")
),
controlbar = dashboardControlbar(id = "controlbar")
),
server = function(input, output, session) {

observeEvent(input$controlbar, {
  if (input$controlbar) {
    showModal(modalDialog(
      title = "Alert",
      "The controlbar is opened.",
      easyClose = TRUE,
      footer = NULL
    ))
  }
})

observeEvent(input$controlbarToggle, {
  updateControlbar("controlbar")
})

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

# controlbar with controlbarMenu
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(),
      controlbar = dashboardControlbar(
        id = "controlbar",
        controlbarMenu(
          id = "menu",
          controlbarItem(
            "Tab 1",
            "Welcome to tab 1"
          ),
          controlbarItem(
            "Tab 2",
            "Welcome to tab 2"
          )
        )
      )
    ),
    server = function(input, output, session) {

      observeEvent(input$control, {
        
      })
    }
  )
}```
showModal(modalDialog(
    title = "Alert",
    sprintf(" %s is active", input$menu),
    easyClose = TRUE,
    footer = NULL
  ))
)
)
)

# Update a controlbar menu
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        radioButtons("controller", "Controller", choices = c(1, 2, 3))
      ),
      controlbar = dashboardControlbar(
        id = "controlbar",
        controlbarMenu(
          id = "menu",
          controlbarItem(
            paste0("Tab", 1),
            paste("Welcome to tab", 1)
          ),
          controlbarItem(
            paste0("Tab", 2),
            paste("Welcome to tab", 2)
          ),
          controlbarItem(
            paste0("Tab", 3),
            paste("Welcome to tab", 3)
          )
        )
      ),
    ),

    server = function(input, output, session) {
      observeEvent(input$controller, {
        updateControlbarMenu("menu",
          selected = paste0("Tab", input$controller)
        )
      })
    }
  )
)


**dashboardFooter**

This creates a dashboard footer

**Usage**

dashboardFooter(left = NULL, right = NULL)

**Arguments**

- *left*: Left text.
- *right*: Right text.

**Examples**

```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(),
      footer = dashboardFooter(
        left = "By Divad Nojnarg",
        right = "Zurich, 2019"
      ),
      title = "DashboardPage"
    ),
    server = function(input, output) { }
  )
}
```

---

**dashboardHeader**

Create a header for a dashboard page

**Description**

A dashboard header can be left blank, or it can include dropdown menu items on the right side.

**Usage**

dashboardHeader(
  ...,
  title = NULL,
  titleWidth = NULL,
  disable = FALSE,
  .list = NULL,
  leftUi = NULL,
  controlbarIcon = shiny::icon("gears"),
  fixed = FALSE
)
Arguments

... Items to put in the header. Should be `dropdownMenus`.

title An optional title to show in the header bar. By default, this will also be used as the title shown in the browser’s title bar. If you want that to be different from the text in the dashboard header bar, set the title in `dashboardPage`.

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

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 Items that will appear on the left part of the navbar. Should be wrapped in a `tagList`.

controlbarIcon Customize the trigger icon of the right sidebar.

fixed Whether the navbar is fixed-top or not. FALSE by default.

Note

We do not recommend to insert shiny input elements (such as sliderInput) in the left menu, since they will not be well displayed. Instead, wrap them in a `dropdownBlock`

See Also

dropdownMenu

Examples

```r
if (interactive()) {
  library(shiny)
  library(shinyWidgets)
  library(shinydashboard)
  library(shinydashboardPlus)
  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(
        leftUi = tagList(
          dropdownBlock(
            id = "mydropdown",
            title = "Dropdown 1",
            icon = icon("sliders"),
            sliderInput(
              inputId = "n",
              label = "Number of observations",
              min = 10, max = 100, value = 30
            ),
            prettyToggle(
              inputId = "na",
              label_on = "NAs kept",
              label_off = "NAs removed",
              icon_on = icon("check"),
              icon_off = icon("remove")
            )
          )
        )
      )
    )
  )
}
```
dashboardLabel

Description
Create a label

Usage
dashboardLabel(..., status, style = "default")

Arguments
... any text.
status label status. Valid statuses are defined as follows:

- primary: #3c8dbc
- success: #00a65a
- info: #00c0ef
- warning: #f39c12
- danger: #f56954

style label border style: "default" (rounded angles), "circle" or "square".

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        dashboardLabel("Label 1", status = "info"),
        dashboardLabel("Label 2", status = "danger", style = "circle"),
        dashboardLabel("Label 3", status = "success", style = "square")
      ),
    ),
    server = function(input, output) { }
  )
}
```

Description

This creates a dashboard page for use in a Shiny app.

Usage

dashboardPage(
  header,
  sidebar,
  body,
  controlbar = NULL,
  footer = NULL,
  title = NULL,
  skin = c("blue", "blue-light", "black", "black-light", "purple", "purple-light",
           "green", "green-light", "red", "red-light", "yellow", "yellow-light", "midnight"),
)
dashboardPage

```r
everything = NULL,
preloader = NULL,
md = FALSE,
options = NULL,
scrollToTop = FALSE
)

Arguments

header A header created by `dashboardHeader`

sidebar A sidebar created by `dashboardSidebar`

body A body created by `dashboardBody`

controller A right sidebar created by `dashboardController`. NULL by default.

footer A footer created by `dashboardFooter`

title A title to display in the browser’s title bar. If no value is provided, it will try to extract the title from the `dashboardHeaderPlus`.

skin A color theme. See https://adminlte.io/themes/AdminLTE/pages/UI/general.html. If the skin is light, the sidebar will have a light background. Not compatible with `freshTheme`.

controller A skin powered by the fresh package. Not compatible with skin. See https://dreamrs.github.io/fresh/articles/vars-shinydashboard.html.

preloader shinydashboardPlus uses waiter (see https://waiter.john-coene.com/#!/). Pass a list like list(html = spin_1(),color = "#333e48"). waiter expects to provide a sub-list to configure `waiterShowOnLoad` (refer to the package help for all styles). duration defines the loader timeout.

md Whether to enable material design. Experimental...

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

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

See Also

dashboardHeader, dashboardSidebar, dashboardBody.

Examples

```
Create a dashboard sidebar.

**Description**

A dashboard sidebar typically contains a `sidebarMenu`, although it may also contain a `sidebarSearchForm`, or other Shiny inputs.

`updateSidebar` allows to toggle a `dashboardSidebar` on the client.

**Usage**

```r
dashboardSidebar(
  ..., 
  disable = FALSE,
  width = NULL,
  collapsed = FALSE,
  minified = TRUE,
  id = NULL
)
```

```r
clearacerb
updateSidebar(id, session = shiny::getDefaultReactiveDomain())
```
Arguments

... Items to put in the sidebar.

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.

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.

id Sidebar id.

session Shiny session object.

Examples

if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  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("sidebar")
      })

      observe({
        print(input$sidebar)
      })
    }
  )
}
dashboardUser  Create a dashboard user profile.

Description

Create a dashboard user profile.

Usage

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

Arguments

...  Body content. Slot for dashboardUserItem.
name  User name.
image  User profile picture.
title  A title.
subtitle  A subtitle.
footer  Footer is any.

See Also

userOutput and renderUser for dynamically-generating dashboardUser.

Examples

if (interactive()) {
  library(shiny)
  library(shinyWidgets)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(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",  
          ...
dashboardUserItem

Create a dashboard user profile item

dashboardUserItem

Description

This can be inserted in a dashboardUser.

Usage

dashboardUserItem(item, width)

Arguments

item HTML Tag.
width Item width between 1 and 12.

dropdownBlock

Create a dropdown block to place in a dashboard header

dropdownBlock

Description

Create a dropdown block to place in a dashboard header

Usage

dropdownBlock(..., id, icon = NULL, title = NULL, badgeStatus = "danger")
Arguments

... Items to put in the menu.

id Dropdown block id.

icon An icon to display in the header. Expect icon.

title Dropdown block title.

badgeStatus Dropdown badge status.

See Also

dashboardHeader for example usage.

flipBox A flipBox based on the W3C example

Description

flipBox creates a box that flips from back to front and inversely
updateFlipBox programatically toggles a flipBox from the server.

Usage

flipBox(id, front, back, trigger = c("click", "hover"), width = 6)

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

Arguments

id flipBox id.

front ui for the front of the flip box

back ui for the back of the flip box

trigger How to trigger rotate effect. Either click or hover. Default to click.

width flipbox width. Between 1 and 12.

session Shiny session object.

Details

To access the state of the flipbox use the input alias input$id$. For example, if your flipBox’s id is myawesomeflipbox, you can access its state via input$myawesomeflipbox where TRUE corresponds to the front, FALSE to the back.
Examples

```r
if (interactive()) {
library(shiny)
library(shinydashboard)
library(shinydashboardPlus)
shinyApp(
  ui = dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
      fluidRow(
        column(
          width = 6,
          uiOutput("active_side"),
          flipBox(
            id = "myflipbox",
            trigger = "hover",
            width = 12,
            front = div(
              class = "text-center",
              h1("Flip on hover"),
              img(
                src = "https://image.flaticon.com/icons/svg/149/149076.svg",
                height = "300px",
                width = "100%"
              )
            ),
            back = div(
              class = "text-center",
              height = "300px",
              width = "100%",
              h1("Flip on hover"),
              p("More information....")
            )
          )
        ),
        column(
          width = 6,
          uiOutput("active_side_2"),
          flipBox(
            id = "myflipbox2",
            width = 12,
            front = div(
              class = "text-center",
              h1("Flip on click"),
              img(
                src = "https://image.flaticon.com/icons/svg/149/149076.svg",
                height = "300px",
                width = "100%"
              )
            ),
            back = div(
              class = "text-center",
              height = "300px",
              width = "100%",
              h1("Flip on click")
            )
          )
        )
      )
    )
  )
}```
server = function(input, output, session) {
  output$active_side <- renderUI({
    side <- if (input$myflipbox) "front" else "back"
    dashboardBadge(side, color = "blue")
  })
  output$active_side_2 <- renderUI({
    side <- if (input$myflipbox2) "front" else "back"
    dashboardBadge(side, color = "blue")
  })
}

if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)
  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        actionButton("toggle", "Toggle flip box"),
        uiOutput("active_side"),
        flipBox(
          id = "myflipbox",
          front = div(
            class = "text-center",
            img(
              src = "https://image.flaticon.com/icons/svg/149/149076.svg",
              height = "300px",
              width = "100%"
            )
          ),
          back = div(
            class = "text-center",
            height = "300px",
            width = "100%",
            h1("Details...."),
            p("More information....")
          )
        )
      ),
    ),
    server = function(input, output, session) {
      output$active_side <- renderUI({
        side <- if (input$myflipbox) "front" else "back"
        dashboardBadge(side, color = "blue")
      })
    }
  )
}
loadingState

AdminLTE2 loading state element

Description

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

Usage

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(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "loading spinner",
          loadingState()
        ),
        title = "Loading State"
      ),
      server = function(input, output) { }
    )
  )
}
messageItem  

Custom messageItem

Description

Custom messageItem

Usage

```r
messageItem(
  from,
  message,
  icon = shiny::icon("user"),
  time = NULL,
  href = NULL,
  inputId = NULL
)
```

Arguments

- `from`: Who the message is from.
- `message`: Text of the message.
- `icon`: An icon tag, created by `icon`.
- `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.
- `href`: An optional URL to link to.
- `inputId`: If not NULL, this item behaves like an action button.

navPills  

AdminLTE2 nav pill container

Description

`navPills` creates a container for nav elements. They are vertically stacked. To insert in `box`.

`updateNavPills` allows to programmatically change the currently selected `navPillsItem` on the client.

`navPillsItem` creates a nav pill item.

Usage

```r
navPills(..., id = NULL)
```

```r
updateNavPills(id, selected, session = shiny::getDefaultReactiveDomain())
```

```r
navPillsItem(
  left = NULL,
)```


```r
right = NULL,
color = NULL,
icon = NULL,
selected = FALSE
)
```

**Arguments**

- `id`  
  navPills unique id to target.
- `selected`  
  Whether the item is active or not. FALSE by default.
- `session`  
  Shiny session object.
- `left`  
  pill left text.
- `right`  
  pill right text.
- `color`  
  pill color: see here for a list of valid colors [https://adminlte.io/themes/AdminLTE/pages/UI/general.html](https://adminlte.io/themes/AdminLTE/pages/UI/general.html). See below:
  - light-blue (primary status): #3c8dbc.
  - red (danger status): #dd4b39.
  - green (success status): #00a65a.
  - aqua (info status): #00c0ef.
  - yellow (warning status): #f39c12.
  - blue: #0073b7.
  - navy: #001F3F.
  - teal: #39CCCC.
  - olive: #5cb85c.
  - lime: #01FF70.
  - orange: #FF851B.
  - fuchsia: #F012BE.
  - purple: #605ca8.
  - maroon: #D81B60.
  - black: #111.
  - gray: #d2d6de.
- `icon`  
  pill icon, if any.

**Author(s)**

David Granjon, <dgranjon@ymail.com>

**Examples**

```r
# navPills
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
```
# update navPills
if (interactive()) {
    library(shiny)
    library(shinydashboard)
    library(shinydashboardPlus)

dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
        radioButtons("controller", "Controller", choices = c(1, 2, 3)),
        br(),
        box(
            title = "Nav Pills",
            status = "info",
            "Box Body",
            footer = navPills(
                id = "pillItem",
                navPillsItem(
                    left = "Item 1",
                    color = "green",
                    right = 10
                ),
                navPillsItem(
                    left = "Item 2",
                    color = "red",
                    icon = icon("angle-down"),
                    right = "10%",
                    active = TRUE
                )
            ),
            title = "Nav Pills"
        ),
        server = function(input, output) {
            observeEvent(input$pillItem, {
                showNotification(sprintf("You clicked on pill N° %s", input$pillItem), type = "message")
            })
        }
    )
)

}
notificationItem

),

navPillsItem(
  left = "Item 2",
  color = "red",
  icon = icon("angle-down"),
  right = "10%"
),

navPillsItem(
  left = "Item 3",
  color = "blue",
  icon = icon("angle-up"),
  right = "30%"
)
)

title = "Nav Pills"

),

server = function(input, output, session) {
  observeEvent(input$controller, {
    updateNavPills(id = "pills", selected = input$controller)
  })
  observeEvent(input$pills, {
    showNotification(sprintf("You selected pill N° %s", input$pills), type = "message")
  })
}


---

**notificationItem**  Custom notificationItem

**Description**

Custom notificationItem

**Usage**

notificationItem(
  text,
  icon = shiny::icon("warning"),
  status = "success",
  href = NULL,
  inputId = NULL
)

**Arguments**

- **text**  The notification text.
- **icon**  An icon tag, created by icon.
- **status**  The status of the item. This determines the item’s background color. Valid statuses are listed in validStatuses.
- **href**  An optional URL to link to.
### Description

`productList` creates a product list container `productList` creates a container to display commercial items in an elegant container. Insert in a box.

### Usage

```r
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: see here for a list of valid colors [https://adminlte.io/themes/AdminLTE/pages/UI/general.html](https://adminlte.io/themes/AdminLTE/pages/UI/general.html). See below:
  - light-blue (primary status): #3c8dbc.
  - red (danger status): #dd4b39.
  - green (success status): #00a65a.
  - aqua (info status): #00c0ef.
  - yellow (warning status): #f39c12.
  - blue: #0073b7.
  - navy: #001f3f.
  - teal: #39CCCC.
  - olive: #3d9970.
  - lime: #01ff70.
  - orange: #ff851b.
  - fuchsia: #f012be.
  - purple: #605ca8.
  - maroon: #d81b60.
  - black: #111.
  - gray: #d2d6de.

### Author(s)

David Granjon, <dgranjon@ymail.com>
Examples

# Box with productList
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  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 = "yellow",
              "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 = "red",
              "This is were I spend most of my time!"
            )
          )
        ),
        title = "Product List"
    ),
    server = function(input, output) { }
  )
}

progressBar

AdminLTE2 vertical progress bar

Description

This creates a vertical progress bar.

Usage

progressBar(
  value,
  min = 0,
  max = 100,
  vertical = FALSE,
progressBar

striped = FALSE,
animated = FALSE,
status = "primary",
size = NULL,
label = NULL
)

Arguments

value Progress bar value. Must be between min and max.
min Progress bar minimum value (0 by default).
max Progress bar maximum value (100 by default).
vertical Progress vertical layout. Default to FALSE.
striped Whether the progress is striped or not. FALSE by default.
animated Whether the progress is active or not. FALSE by default. Works only if striped is TRUE.
status Progress bar status. "primary" by default or "warning", "info", "danger" or "success". Valid statuses are defined as follows:
  • primary: #3c8dbc
  • success: #00a65a
  • info: #00c0ef
  • warning: #f39c12
  • danger: #f56954
size Progress bar size. NULL by default: "sm", "xs" or "xxs" also available.
label Progress label. NULL by default.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(
        box(
          title = "Horizontal",
          progressBar(
            value = 10,
            striped = TRUE,
            animated = TRUE,
            label = "10%"
          ),
          progressBar(
        )
      )
    )
  )
}
renderUser

Create dynamic user output (server side)

Description
Create dynamic user output (server side)

Usage
renderUser(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 environment in which to evaluate `expr`. 
quoted Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

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

userOutput for the corresponding client side function and examples.

Other user outputs: userOutput()

---

### shinydashboardPlus

**Description**

Extend 'shinydashboard' with 'AdminLTE2' components. 'AdminLTE2' is a free 'Bootstrap 3' dashboard template available at <https://adminlte.io>. Customize boxes, add timelines and a lot more.

**Author(s)**

**Maintainer**: David Granjon <dgranjon@ymail.com>

Other contributors:

- RinteRface [copyright holder]
- Almasaeed Studio (AdminLTE2 theme for Bootstrap 3) [contributor, copyright holder]
- Guang Yang (ygdashboard original template) [contributor, copyright holder]
- Winston Chang (Functions from shinydashboard) [contributor, copyright holder]
- Victor Perrier <victor.perrier@dreamrs.fr> (improved the shinydashboardPlusGallery) [contributor]

**See Also**

Useful links:

- [https://github.com/RinteRface/shinydashboardPlus](https://github.com/RinteRface/shinydashboardPlus)
- [https://rinterface.com/shiny/shinydashboardPlus/](https://rinterface.com/shiny/shinydashboardPlus/)
Launch the shinydashboardPlus Gallery

Description

A gallery of all components available in shinydashboardPlus.

Usage

shinydashboardPlusGallery()

Examples

if (interactive()) {
  shinydashboardPlusGallery()
}

skinSelector

AdminLTE2 skin selector

Description

This creates a skin selector element.

Usage

skinSelector()

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      header = dashboardHeader(),
      sidebar = dashboardSidebar(),
      body = dashboardBody(),
      controlbar = dashboardControlbar(skinSelector()),
      title = "Skin Selector"
    ),
    server = function(input, output) { }
  )
}
skinSelectorItem  
*AdminLTE2 skin selector item*

**Description**

This creates an item to insert in a skin selector object.

**Usage**

```r
skinSelectorItem(color)
```

**Arguments**

- `color`  

**Author(s)**

David Granjon, <dgranjon@ymail.com>

---

socialBox  
*AdminLTE2 social box*

**Description**

`socialBox` creates a special box dedicated for social content.

- `userBlock` goes in the title of `socialBox`.
- `boxComment` has to be inserted in the comment slot of `socialBox`.

**Usage**

```r
socialBox(
  ..., 
  title = NULL,
  footer = NULL,
  width = 6,
  height = NULL,
  collapsible = TRUE,
  collapsed = FALSE,
  closable = FALSE,
  box ToolSize = "sm",
  headerBorder = TRUE,
  label = NULL,
  dropdownMenu = NULL,
  sidebar = NULL,
  id = NULL
)

userBlock(image, title, subtitle = NULL)

boxComment(..., image, title = NULL, date = NULL)
```
socialBox

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 valueWidth 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.
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.
boxToolSize Size of the toolbox: choose among "xs", "sm", "md", "lg".
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 If passed, the item will behave like an action button.
image author image, if any.
subtitle Any subtitle.
date date of publication.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if (interactive()) {
library(shiny)
library(shinydashboard)
library(shinydashboardPlus)

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/AdminLTE/dist/img/photo1.png",
...
socialButton

AdminLTE2 social button

Description
Create a social button

Usage
socialButton(href, icon)

Arguments
- href: External link.

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

Examples
```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(
```
starBlock

```r
dashboardSidebar(),
dashboardBody(
  box(
    title = "Social Buttons",
    status = NULL,
    socialButton(
      href = "https://dropbox.com",
      icon = icon("dropbox"),
    ),
    socialButton(
      href = "https://github.com",
      icon = icon("github")
    )
  ),
  title = "Social Buttons"
),
server = function(input, output) { }
)
```

---

starBlock  

AdminLTE2 starBlock

### Description

Create a starBlock item (ideal for rating)

### Usage

`starBlock(value, max = 5, color = "yellow")`

### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Current score. 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: see <code>validColors()</code> in the documentation. See below:</td>
</tr>
<tr>
<td></td>
<td>- light-blue (primary status): #3c8dbc.</td>
</tr>
<tr>
<td></td>
<td>- red (danger status): #dd4b39.</td>
</tr>
<tr>
<td></td>
<td>- green (success status): #00a65a.</td>
</tr>
<tr>
<td></td>
<td>- aqua (info status): #00c0ef.</td>
</tr>
<tr>
<td></td>
<td>- yellow (warning status): #f39c12.</td>
</tr>
<tr>
<td></td>
<td>- blue: #0073b7.</td>
</tr>
<tr>
<td></td>
<td>- navy: #001f3f.</td>
</tr>
<tr>
<td></td>
<td>- teal: #39CCCC.</td>
</tr>
<tr>
<td></td>
<td>- olive: #3d9970.</td>
</tr>
<tr>
<td></td>
<td>- lime: #01ff70.</td>
</tr>
<tr>
<td></td>
<td>- orange: #ff851b.</td>
</tr>
<tr>
<td></td>
<td>- fuchsia: #f012be.</td>
</tr>
</tbody>
</table>
• purple: #605ca8.
• maroon: #D81B60.
• black: #111.
• gray: #d2d6de.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

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

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

**Arguments**

- `text` The task text.
- `value` A percent value to use for the bar.
- `color` A color for the bar. Valid colors are listed in `validColors`.
- `href` An optional URL to link to.
- `inputId` If not NULL, this item behaves like an action button.
```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(
        dropdownMenu(
          type = "tasks",
          badgeStatus = "danger",
          taskItem(
            inputId = "mytask",
            value = 20,
            color = "aqua",
            text = "Click me!"
          ),
          taskItem(
            value = 40,
            color = "green",
            text = "Basic item"
          )
        ),
        taskItem()
      ),
      dashboardSidebar(),
      dashboardBody(),
      title = "Dashboard example"
    ),
    server = function(input, output) {
      observeEvent(input$mytask, {
        showModal(modalDialog(
          title = "Important message",
          "This is an important message!"
        ))
      })
    }
  )
```

### 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

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

timelineLabel(..., color = NULL)

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

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

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

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

Arguments

... any element.
reversed Whether the timeline is reversed or not.
width media width in pixels.
color item color: see here for a list of valid colors https://adminlte.io/themes/AdminLTE/pages/UI/general.html. See below:
  • light-blue (primary status): #3c8dbc.
  • red (danger status): #dd4b39.
  • green (success status): #00a65a.
  • aqua (info status): #00c0ef.
  • yellow (warning status): #f39c12.
  • blue: #0073b7.
  • navy: #001F3F.
  • teal: #39CCCC.
  • olive: #3D9970.
  • lime: #01FF70.
  • orange: #FF851B.
  • fuchsia: #F012BE.
  • purple: #605ca8.
  • maroon: #D81B60.
  • black: #111.
  • gray: #d2d6de.
icon item icon. Expect icon.
time item date or time.
title item title.
timelineBlock

border  Whether to display a border between the header and the body. TRUE by default.
footer  item footer if any.
image   media url or path.
height  media height in pixels.

Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)
  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        h3("When Reversed = TRUE, can be displayed inside a box"),
        box(
          title = "Timeline",
          status = "info",
          timelineBlock(
            width = 12,
            timelineEnd(color = "red"),
            timelineLabel(2018, color = "teal"),
            timelineItem(
              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(2015, color = "orange"),
            timelineItem(
              title = "Item 3",
              icon = icon("paint-brush"),
              color = "maroon",
              timelineItemMedia(image = "https://placehold.it/150x100"),
              timelineItemMedia(image = "https://placehold.it/150x100")
            ),
            timelineStart(color = "purple")
          ),
        )
      )
    )
  )
  h3("When Reversed = FALSE, can be displayed out of a box")
  timelineBlock(
    reversed = FALSE,
    timelineEnd(color = "red"),
  )
)
todoList

AdminLTE2 todo list container

description

Create a todo list container. May be included in box.

todoListItem creates a todo list item.

Usage

todoList(..., sortable = TRUE)

todoListItem(..., checked = FALSE, label = NULL)

Arguments

... any element such as labels, ...
sortable Whether the list elements are sortable or not.
checked Whether the list item is checked or not.
label item label.
Author(s)

David Granjon, <dgranjon@ymail.com>

Examples

```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinyjqui)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          "Sortable todo list demo",
          status = "warning",
          todoList(
            todoListItem(
              label = "Design a nice theme",
              "Some text here"
            ),
            todoListItem(
              label = "Make the theme responsive",
              "Some text here"
            ),
            todoListItem(
              checked = TRUE,
              label = "Let theme shine like a star"
            )
          ),
          box(
            "Simple todo list demo",
            status = "warning",
            todoList(
              sortable = FALSE,
              todoListItem(
                label = "Design a nice theme",
                "Some text here"
              ),
              todoListItem(
                label = "Make the theme responsive",
                "Some text here"
              ),
              todoListItem(
                checked = TRUE,
                label = "Let theme shine like a star"
              )
            ),
            title = "Todo Lists"
          )
        )
      ),
      title = "Todo Lists"
    ),
  )
```

server = function(input, output) { }
}

userBox

AdminLTE2 user box

Description

userBox creates a user card.
userDescription creates a customized title tag for userBox.

Usage

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

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

Arguments

... any element such as descriptionBlock.
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: #3c8dbc
• success: #00a65a
• info: #00c0ef
• warning: #f39c12
• danger: #f56954
• navy: #001F3F
• teal: #39CCCC
• purple: #605ca8
• orange: #ff851b
• maroon: #D81B60
• black: #111111

Only primary, success, info, warning and danger are compatible with solid-Header!

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:
• light-blue (primary status): #3c8dbc.
• red (danger status): #dd4b39.
• green (success status): #00a65a.
• aqua (info status): #00c0ef.
• yellow (warning status): #f39c12.
• blue: #0073b7.
• navy: #001F3F.
• teal: #39CCCC.
• olive: #3D9970.
• lime: #01FF70.
• orange: #FF851B.
• fuchsia: #F012BE.
• purple: #605ca8.
• maroon: #D81B60.
• black: #111.
• gray: #d2d6de.

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 valueBox 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.

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.

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

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

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

label
Slot for boxLabel.
userBox

dropdownMenu List of items in the boxtool dropdown menu. Use boxDropdown.
sidebar Slot for boxSidebar.
id If passed, the item will behave like an action button.
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>

Examples

if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  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 = "light-blue",
          boxToolSize = "xl",
          "Some text here!",
          "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 = "purple",
          closable = TRUE,
          "Some text here!"
        )
      )
    )
  )
}
userList

AdminLTE2 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

```r
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(
        box(
          title = "User List example",
          status = "success",
          userList(
            userListItem(
              image = "https://adminlte.io/themes/AdminLTE/dist/img/user1-128x128.jpg",
              title = "Shiny",
              subtitle = "Package 1"
            ),
            userListItem(
              image = "https://adminlte.io/themes/AdminLTE/dist/img/user7-128x128.jpg",
              title = "Tidyverse",
              subtitle = "Package 2"
            ),
            userListItem(
              image = "https://adminlte.io/themes/AdminLTE/dist/img/user5-128x128.jpg",
              title = "tidyr",
              subtitle = "Package 3"
            )
          )
        ),
        title = "User List"
      ),
      server = function(input, output) { }
    )
  )
}
```

userMessages

---

**AdminLTE2 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,
    date = NULL,
    image = NULL,
    type = c("sent", "received")
)

updateUserMessages(
    id,
    action = c("add", "remove", "update"),
    index = NULL,
    content = NULL,
    session = shiny::getDefaultReactiveDomain()
)

Arguments

... Message text.
id userMessages to target.
status Messages status. See here for a list of valid colors [https://adminlte.io/themes/AdminLTE/pages/UI/general.html](https://adminlte.io/themes/AdminLTE/pages/UI/general.html). Valid statuses are defined as follows:
  • primary: #3c8dbc
  • success: #00a65a
  • info: #00c0ef
  • warning: #f39c12
  • danger: #f56954
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>
if (interactive()) {
library(shiny)
library(shinydashboard)
library(shinydashboardPlus)

shinyApp(
  ui = dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    dashboardBody(
      box(
        title = "Box with messages",
        solidHeader = TRUE,
        status = "warning",
        userMessages(
          width = 12,
          status = "success",
          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 = "sent",
            "You better believe it!"
          )
        ),
        status = "danger",
        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!"
          )
        ),
        title = "User Messages"
      ),
      server = function(input, output) { }
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  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/f115dfc9cab063597b122d015996b39.jpg",
            type = "received",
            text = tagList(}
userOutput

Create a dynamic user output (client side)

Description

This can be used as a placeholder for dynamically-generated dashboardUser.

Usage

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

Arguments

id       Output variable name.
tag      A tag function, like tags$li or tags$ul.

See Also

renderUser for the corresponding server side function and examples.
Other user outputs: renderUser()

userPost  AdminLTE2 user post

Description

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

Usage

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

userPostTagItems(...)

userPostTagItem(..., side = "left")

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

Arguments

... tool content such as label, button, ...
id     unique id of the post.
image  image path or url ...
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.
side  tool item orientation: "left" of "right", "left" by default.
height  media height in pixels.
width  media width in pixels.

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

Examples
if (interactive()) {
  library(shiny)
  library(shinydashboard)
  library(shinydashboardPlus)

  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(dashboardLabel("item 1", status = "info")),
            userPostTagItem(dashboardLabel("item 2", status = "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(dashboardLabel("item 1", status = "success")),
          userPostTagItem(dashboardLabel("item 2", status = "danger"), side = "right")
        )
      ),
      title = "userPost"
    ),
    server = function(input, output) { }
  )
}
Index

accordion, 2, 2
accordionItem, 2, 3
accordionItem (accordion), 2
actionButton, 21
appButton, 5
attachmentBlock, 6, 6

blockQuote, 7
box, 6, 8, 8, 17, 19, 40, 44, 55, 58, 63, 64, 69
boxComment, 50
boxComment (socialBox), 50
boxDropdown, 8, 10, 51, 62
boxDropdown (box), 8
boxDropdownItem, 8
boxDropdownItem (box), 8
boxLabel, 10, 17, 17, 51, 61
boxPad, 8
boxPad (box), 8
boxProfileItem, 17, 18
boxProfileItem (boxProfile), 17
boxSidebar, 10, 19, 19, 51, 62
carousel, 20, 20
carouselItem, 20
carouselItem (carousel), 20
controlbarItem, 23, 24
controlbarItem (dashboardControlbar), 22
controlbarMenu, 23
controlbarMenu (dashboardControlbar), 22
dashboardBadge, 21
dashboardBody, 31
dashboardControlbar, 22, 23, 31
dashboardFooter, 26, 31
dashboardHeader, 27, 31, 36
dashboardLabel, 29
dashboardPage, 28, 30
dashboardSidebar, 21, 31, 32, 32
dashboardUser, 34, 34, 35, 68
dashboardUserItem, 34, 35
descriptionBlock, 8, 9, 60
descriptionBlock (box), 8
dropdownBlock, 28, 35
dropdownDivider, 8
dropdownDivider (box), 8
dropdownMenu, 28
flipBox, 36, 36

HTML(), 47
icon, 10, 12, 19, 36, 40, 43, 56
icon(), 5
lapply, 20
loadingState, 39
messageItem, 40

navbarPage(), 24
navPills, 40, 40, 41
navPillsItem, 40, 41
navPillsItem (navPills), 40
notificationItem, 43

productList, 44, 44
productListItem, 44
productListItem (productList), 44
progressBar, 45

renderUser, 34, 47, 69

shinydashboardPlus, 48
shinydashboardPlus-package
(shinydashboardPlus), 48

shinydashboardPlusGallery, 49

sidebarMenu, 32
sidebarSearchForm, 32

skinSelector, 49
skinSelectorItem, 50

socialBox, 50, 50

socialButton, 52

starBlock, 53
taskItem, 54
timelineBlock, 55, 55
timelineEnd, 55
timelineEnd (timelineBlock), 55
INDEX

timelineItem, 55
timelineItem (timelineBlock), 55
timelineItemMedia, 55
timelineItemMedia (timelineBlock), 55
timelineLabel, 55
timelineLabel (timelineBlock), 55
timelineStart, 55
timelineStart (timelineBlock), 55
todoList, 58
todoListItem, 58
todoListItem (todoList), 58
uiOutput(), 48
updateAccordion, 2
updateAccordion (accordion), 2
updateBox, 8
updateBox (box), 8
updateBoxSidebar, 19
updateBoxSidebar (boxSidebar), 19
updateControlbar, 23
updateControlbar (dashboardControlbar), 22
updateControlbarMenu, 23
updateControlbarMenu (dashboardControlbar), 22
updateFlipBox, 36
updateFlipBox (flipBox), 36
updateNavPills, 40
updateNavPills (navPills), 40
updateSidebar, 32
updateSidebar (dashboardSidebar), 32
updateUserMessages, 64
updateUserMessages (userMessages), 64
userBlock, 50
userBlock (socialBox), 50
userBox, 60, 60
userDescription, 60
userDescription (userBox), 60
userList, 63, 63
userListItem, 63
userListItem (userList), 63
userMessage, 64
userMessage (userMessages), 64
userMessages, 64, 64, 65
userOutput, 34, 48, 68
userPost, 69, 69
userPostMedia, 69
userPostMedia (userPost), 69
userPostTagItem, 69
userPostTagItem (userPost), 69
userPostTagItems, 69
userPostTagItems (userPost), 69
validateCssUnit(), 5
validColors, 10, 54, 61
validStatuses, 43
waiterShowOnLoad, 31