Package ‘shinydashboard’

October 14, 2022

Title Create Dashboards with 'Shiny'
Version 0.7.2
Description Create dashboards with 'Shiny'. This package provides a theme on top of 'Shiny', making it easy to create attractive dashboards.

URL http://rstudio.github.io/shinydashboard/
Depends R (>= 3.0)
License GPL (>= 2) | file LICENSE
Imports utils, shiny (>= 1.0.0), htmltools (>= 0.2.6), promises
BugReports https://github.com/rstudio/shinydashboard
RoxygenNote 6.0.1.9000
NeedsCompilation no
Author Winston Chang [aut, cre],
Barbara Borges Ribeiro [aut],
RStudio [cph],
Almasaeed Studio [ctb, cph] (AdminLTE theme for Bootstrap),
Adobe Systems Incorporated [ctb, cph] (Source Sans Pro font)
Maintainer Winston Chang <winston@rstudio.com>
Repository CRAN
Date/Publication 2021-09-30 12:10:06 UTC

R topics documented:

<table>
<thead>
<tr>
<th>topic</th>
<th>pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>box</td>
<td>2</td>
</tr>
<tr>
<td>dashboardBody</td>
<td>5</td>
</tr>
<tr>
<td>dashboardHeader</td>
<td>6</td>
</tr>
<tr>
<td>dashboardPage</td>
<td>8</td>
</tr>
<tr>
<td>dashboardSidebar</td>
<td>9</td>
</tr>
<tr>
<td>dropdownMenu</td>
<td>10</td>
</tr>
<tr>
<td>dropdownMenuOutput</td>
<td>11</td>
</tr>
<tr>
<td>infoBox</td>
<td>11</td>
</tr>
<tr>
<td>menuItemOutput</td>
<td>12</td>
</tr>
</tbody>
</table>
Create a box for the main body of a dashboard

Description

Boxes can be used to hold content in the main body of a dashboard.

Usage

```r
box(..., title = NULL, footer = NULL, status = NULL,
    solidHeader = FALSE, background = NULL, width = 6, height = NULL,
    collapsible = FALSE, collapsed = FALSE)
```

Arguments

- `...`: Contents of the box.
- `title`: Optional title.
- `footer`: Optional footer text.
- `status`: The status of the item. This determines the item's background color. Valid statuses are listed in `validStatuses`.
- `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`. 

Index

- `box` 27
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.

See Also

Other boxes: infoBox, tabBox, valueBox

Examples

```r
## Only run this example in interactive R sessions
if (interactive()) {
  library(shiny)

  # A dashboard body with a row of infoBoxes and valueBoxes, and two rows of boxes
  body <- dashboardBody(

  # infoBoxes
  fluidRow(
    infoBox(
      "Orders", uiOutput("orderNum2"), "Subtitle", icon = icon("credit-card"))
    ),
    infoBox(
      "Approval Rating", "60\%", icon = icon("line-chart"), color = "green",
      fill = TRUE
    ),
    infoBox(
      "Progress", uiOutput("progress2"), icon = icon("users"), color = "purple"
    ),
  ),

  # valueBoxes
  fluidRow(
    valueBox(
      uiOutput("orderNum"), "New Orders", icon = icon("credit-card"),
      href = "http://google.com"
    ),
    valueBox(
      tagList("60", tags$sup(style="font-size: 20px", "%")),
      "Approval Rating", icon = icon("line-chart"), color = "green"
    ),
    valueBox(
      htmlOutput("progress"), "Progress", icon = icon("users"), color = "purple"
    ),
  ),
}
```

# Boxes

```r
fluidRow(
  box(status = "primary",
       sliderInput("orders", "Orders", min = 1, max = 2000, value = 650),
       selectInput("progress", "Progress",
                   choices = c("0%" = 0, "20%" = 20, "40%" = 40, "60%" = 60, "80%" = 80, "100%" = 100))
  ),
  box(title = "Histogram box title",
       status = "warning", solidHeader = TRUE, collapsible = TRUE,
       plotOutput("plot", height = 250)
  ),
)
```

# Boxes with solid color, using 'background'

```r
fluidRow(
  # Box with textOutput
  box(
    title = "Status summary",
    background = "green",
    width = 4,
    textOutput("status")
  ),
  # Box with HTML output, when finer control over appearance is needed
  box(
    title = "Status summary 2",
    width = 4,
    background = "red",
    uiOutput("status2")
  ),
  box(
    width = 4,
    background = "light-blue",
    p("This is content. The background color is set to light-blue")
  )
)
```

server <- function(input, output) {
  output$orderNum <- renderText({
    prettyNum(input$orders, big.mark="","
  })

  output$orderNum2 <- renderText({
    prettyNum(input$orders, big.mark="","
  })

  output$progress <- renderUI({
    tagList(input$progress, tags$sup(style="font-size: 20px", "%"))
  })
}
The main body of a dashboard page.

**Description**

The main body typically contains boxes. Another common use pattern is for the main body to contain tabItems.

**Usage**

dashboardBody(...)  

**Arguments**

...  

Items to put in the dashboard body.
See Also

`tabItems, box, valueBox`.

---

**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**

```r
dashboardHeader(..., title = NULL, titleWidth = NULL, disable = FALSE, .list = NULL)
```

**Arguments**

- `...`: Items to put in the header. Should be `dropdownMenu`.
- `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.

**See Also**

`dropdownMenu`

**Examples**

```r
## Only run this example in interactive R sessions
if (interactive()) {
  library(shiny)

  # A dashboard header with 3 dropdown menus
  header <- dashboardHeader(
    title = "Dashboard Demo",
  
  
  # Dropdown menu for messages
  dropdownMenu(type = "messages", badgeStatus = "success",
               messageItem("Support Team",
                           "This is the content of a message.",
                           time = "5 mins")
```
shinyApp(
  ui = dashboardPage(
    header,
    dashboardSidebar(),
    dashboardBody()
  ),
  server = function(input, output) { }
)
### dashboardPage

#### Description

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

#### Usage

```r
dashboardPage(header, sidebar, body, title = NULL, skin = c("blue", "black", "purple", "green", "red", "yellow"))
```

#### Arguments

- **header**: A header created by `dashboardHeader`.
- **sidebar**: A sidebar created by `dashboardSidebar`.
- **body**: A body created by `dashboardBody`.
- **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 `dashboardHeader`.
- **skin**: A color theme. One of “blue”, “black”, “purple”, “green”, “red”, or “yellow”.

#### See Also

- `dashboardHeader`, `dashboardSidebar`, `dashboardBody`.

#### Examples

```r
## Only run this example in interactive R sessions
if (interactive()) {
  # Basic dashboard page template
  library(shiny)
  shinyApp(
    ui = dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody(),
      title = "Dashboard example"
    ),
    server = function(input, output) { }
  )
}
```
Create a dashboard sidebar.

Description

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

Usage

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

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.

See Also

sidebarMenu

Examples

```r
# Only run this example in interactive R sessions
if (interactive()) {
  header <- dashboardHeader()

  sidebar <- dashboardSidebar(
    sidebarUserPanel("User Name",
      subtitle = a(href = ", icon("circle", class = "text-success", "Online"),
      # Image file should be in www/ subdir
      image = "userimage.png" ),
    sidebarSearchForm(label = "Enter a number", "searchText", "searchButton"),
    sidebarMenu(
      # Setting id makes input$tabs give the tabName of currently-selected tab
      id = "tabs",
      menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),
      menuItem("Widgets", icon = icon("th"), tabName = "widgets", badgeLabel = "new",
        badgeColor = "green"),
      menuItem("Charts", icon = icon("bar-chart-o"),
        menuSubItem("Sub-item 1", tabName = "subitem1"),
        menuSubItem("Sub-item 2", tabName = "subitem2")
    )
  )
}
```
dropdownMenu

Create a dropdown menu to place in a dashboard header

Description

Create a dropdown menu to place in a dashboard header

Usage

dropdownMenu(..., type = c("messages", "notifications", "tasks"),
            badgeStatus = "primary", icon = NULL, headerText = NULL, .list = 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 listed in validStatuses. A value of NULL means to not display a badge.
icon An icon to display in the header. By default, the icon is automatically selected depending on type, but it can be overridden with this argument.
dropdownMenuOutput

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 programmatical ly generated items.

See Also

dashboardHeader for example usage.

infoBox  Create an info box for the main body of a dashboard.

Description

An info box displays a large icon on the left side, and a title, value (usually a number), and an optional smaller subtitle on the right side. Info boxes are meant to be placed in the main body of a dashboard.
menuItemOutput

Usage

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

Arguments

- `title`: Title text.
- `value`: The value to display in the box. Usually a number or short text.
- `subtitle`: Subtitle text (optional).
- `icon`: An icon tag, created by `icon`.
- `color`: A color for the box. Valid colors are listed in `validColors`.
- `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.
- `href`: 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.

See Also

- `box` for usage examples.
- Other boxes: `box, tabBox, valueBox`

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`
Create a dynamic menu output for shinydashboard (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

menuOutput(outputId, tag = 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

Create a message item to place in a dropdown message menu

Description

Create a message item to place in a dropdown message menu

Usage

messageItem(from, message, icon = shiny::icon("user"), time = NULL, href = 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.
See Also

dashboardHeader for example usage.
Other menu items: notificationItem, taskItem

---

notificationItem

Create a notification item to place in a dropdown notification menu

Description

Create a notification item to place in a dropdown notification menu

Usage

notificationItem(text, icon = shiny::icon("warning"), status = "success",
href = 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 sta-
tutes are listed in validStatuses.
href An optional URL to link to.

See Also

dashboardHeader for example usage.
Other menu items: messageItem, taskItem

---

renderDropdownMenu

Create a dropdown menu output (server side; deprecated)

Description

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

Usage

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

Arguments

texr An expression that returns a Shiny tag object, HTML, or a list of such objects.
menv 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.
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 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
- `menuOutput` for the corresponding client side function and examples.

Other menu outputs: dropdownMenuOutput, menuItemOutput, menuOutput, sidebarMenuOutput

Examples
## Only run these examples in interactive R sessions

```r
if (interactive()) {
library(shiny)
# ========= Dynamic sidebarMenu =========
ui <- dashboardPage(
  dashboardHeader(title = "Dynamic sidebar"),
  dashboardSidebar(
    sidebarMenuOutput("menu")
  ),
  dashboardBody()
)

server <- function(input, output) {
  output$menu <- renderMenu({
    sidebarMenu(
      menuItem("Menu item", icon = icon("calendar"))
    )
  })
}
}
```
shinyApp(ui, server)

# =========== Dynamic dropdownMenu ==========
# Example message data in a data frame
messageData <- data.frame(
    from = c("Administrator", "New User", "Support"),
    message = c(
        "Sales are steady this month.",
        "How do I register?",
        "The new server is ready."
    ),
    stringsAsFactors = FALSE
)

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)
    })

dashboardPage()
renderValueBox

Description
This is the server-side function for creating a dynamic valueBox or infoBox.

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

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.

See Also
valueBoxOutput for the corresponding UI-side function.

Examples
## Only run this example in interactive R sessions
if (interactive()) {
  library(shiny)

  ui <- dashboardPage(
    dashboardHeader(title = "Dynamic boxes"),
    dashboardSidebar(),
    dashboardBody(
      fluidRow(
        box(width = 2, actionButton("count", "Count")),
        infoBoxOutput("ibox"),
        valueBoxOutput("vbox")
      )
    )
  )

  server <- function(input, output) {
    output$ibox <- renderInfoBox({
      infoBox(
        "Title",
        input$count,
        icon = icon("credit-card")
      )
    })
    output$vbox <- renderValueBox({
      valueBox(
        "Title",
        input$count,
        icon = icon("credit-card")
      )
    })
  }
}
Description

A dashboardSidebar can contain a sidebarMenu. A sidebarMenu contains menuItems, and they can in turn contain menuSubItems.

Usage

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

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

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

Arguments

... For menu items, this may consist of menuSubItems.

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 tab. If you want to be able to bookmark and restore the selected tab, an id is required.

.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 programatically generated items.

text Text to show for the menu item.
### Details

Menu items (and similarly, sub-items) should have a value for either `href` or `tabName`; otherwise the item would do nothing. If it has a value for `href`, then the item will simply be a link to that value.

If a `menuItem` has a non-NULL `tabName`, then the `menuItem` will behave like a tab – in other words, clicking on the `menuItem` will bring a corresponding `tabItem` to the front, similar to a `tabPanel`. One important difference between a `menuItem` and a `tabPanel` is that, for a `menuItem`, you must also supply a corresponding `tabItem` with the same value for `tabName`, whereas for a `tabPanel`, no `tabName` is needed. (This is because the structure of a `tabPanel` is such that the tab name can be automatically generated.) Sub-items are also able to activate `tabItems`.

Menu items (but not sub-items) also may have an optional badge. A badge is a colored oval containing text.

### See Also

- `dashboardSidebar` for example usage. For dynamically-generated sidebar menus, see `renderMenu` and `sidebarMenuOutput`.
- Other sidebar items: `sidebarSearchForm`, `sidebarUserPanel`

---

**sidebarMenuOutput**

Create a sidebar menu output (client side)

### Description

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

### Usage

```r
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

---

sidebarSearchForm  Create a search form to place in a sidebar

Description

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

Usage

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

Arguments

textId  Shiny input ID for the text input box.
buttonId  Shiny input ID for the search button (which functions like an actionButton).
label  Text label to display inside the search box.
icon  An icon tag, created by icon.

See Also

dashboardSidebar for example usage.

Other sidebar items: sidebarMenu, sidebarUserPanel
sidebarUserPanel

A panel displaying user information in a sidebar

Description
A panel displaying user information in a sidebar

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

Arguments
name Name of the user.
subtitle Text or HTML to be shown below the name.
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.

See Also
dashboardSidebar for example usage.
Other sidebar items: sidebarMenu, sidebarSearchForm

---

tabBox

Create a tabbed box

Description
Create a tabbed box

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

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.
title Title for the tabBox.
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.

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

side  Which side of the box the tabs should be on ("left" or "right"). When side="right", the order of tabs will be reversed.

See Also
Other boxes: box, infoBox, valueBox

Examples

```r
## Only run this example in interactive R sessions
if (interactive()) {
  library(shiny)
  body <- dashboardBody(
    fluidRow(
      tabBox(
        title = "First tabBox",
        # The id lets us use input$tabset1 on the server to find the current tab
        id = "tabset1", height = "250px",
        tabPanel("Tab1", "First tab content"),
        tabPanel("Tab2", "Tab content 2")
      ),
      tabBox(
        side = "right", height = "250px",
        selected = "Tab3",
        tabPanel("Tab1", "Tab content 1"),
        tabPanel("Tab2", "Tab content 2"),
        tabPanel("Tab3", "Note that when side=right, the tab order is reversed."
      ),
      fluidRow(
        tabBox(
          # Title can include an icon
          title = tagList(shiny::icon("gear"), "tabBox status"),
          tabPanel("Tab1",
            "Currently selected tab from first box:",
           verbatimTextOutput("tabset1Selected")
          ),
          tabPanel("Tab2", "Tab content 2")
        )
      )
    ),
    ui = dashboardPage(dashboardHeader(title = "tabBoxes"), dashboardSidebar(), body),
  )
}```
server = function(input, output) {
  # The currently selected tab from the first box
  output$tabset1Selected <- renderText(
    input$tabset1
  )
}

---

**tabItem**

One tab to put inside a tab items container

---

**Description**

One tab to put inside a tab items container

**Usage**

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

**Arguments**

- `tabName` The name of a tab. This must correspond to the `tabName` of a `menuItem` or `menuSubItem`.
- `...` Contents of the tab.

**See Also**

`menuItem`, `menuSubItem`, `tabItems`. See `sidebarMenu` for a usage example.

---

**tabItems**

A container for tab items

---

**Description**

A container for tab items

**Usage**

```r
tabItems(...)```

**Arguments**

- `...` Items to put in the container. Each item should be a `tabItem`.

**See Also**

`menuItem`, `menuSubItem`, `tabItem`. See `sidebarMenu` for a usage example.
taskItem  

Create a task item to place in a dropdown task menu

Description
Create a task item to place in a dropdown task menu

Usage

```
taskItem(text, value = 0, color = "aqua", href = 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.

See Also

- `dashboardHeader` for example usage.
- Other menu items: `messageItem`, `notificationItem`

updateTabItems  

Change the selected tab on the client

Description
This function controls the active tab of `tabItems` from the server. It behaves just like `updateTabsetPanel`.

Usage

```
updateTabItems(session, inputId, selected = NULL)
```

Arguments

- **session**: The `session` object passed to function given to `shinyServer`.
- **inputId**: The id of the `tabsetPanel`, `navlistPanel`, or `navbarPage` object.
- **selected**: The name of the tab to make active.
Examples

```r
## Only run this example in interactive R sessions
if (interactive()) {
  ui <- dashboardPage(
    dashboardHeader(title = "Simple tabs"),
    dashboardSidebar(
      sidebarMenu(
        id = "tabs",
        menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),
        menuItem("Widgets", tabName = "widgets", icon = icon("th")),
      ),
      actionButton('switchtab', 'Switch tab')
    ),
    dashboardBody(
      tabItems(
        tabItem(tabName = "dashboard",
          h2("Dashboard tab content"))
        ),
        tabItem(tabName = "widgets",
          h2("Widgets tab content"))
      )
    )
  )

  server <- function(input, output, session) {
    observeEvent(input$switchtab, {
      newtab <- switch(input$tabs,
        "dashboard" = "widgets",
        "widgets" = "dashboard"
      )
      updateTabItems(session, "tabs", newtab)
    })
  }

  shinyApp(ui, server)
}
```

---

valueBox Create a value box for the main body of a dashboard.

Description

A value box displays a value (usually a number) in large text, with a smaller subtitle beneath, and a large icon on the right side. Value boxes are meant to be placed in the main body of a dashboard.

Usage

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

Arguments

value The value to display in the box. Usually a number or short text.
subtitle Subtitle text.
icont An icon tag, created by icon.
color A color for the box. Valid colors are listed in validColors.
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.
href An optional URL to link to.

See Also

box for usage examples.
Other boxes: box, infoBox, tabBox

valueBoxOutput Create an info or value box output (client side)

Description

This is the UI-side function for creating a dynamic valueBox or infoBox.

Usage

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

Arguments

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 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.

See Also

renderValueBox for the corresponding server-side function and examples.
Index

* boxes
  box, 2
  infoBox, 11
  tabBox, 21
  valueBox, 25

* menu items
  messageItem, 13
  notificationItem, 14
  taskItem, 24

* menu outputs
  dropdownMenuOutput, 11
  menuItemOutput, 12
  menuOutput, 13
  renderMenu, 15
  sidebarMenuOutput, 19

* sidebar items
  sidebarMenu, 18
  sidebarSearchForm, 20
  sidebarUserPanel, 21

ActionButton, 20
box, 2, 5, 6, 12, 22, 26
dashboardBody, 5, 8
dashboardHeader, 6, 8, 11, 14, 24
dashboardPage, 6, 8
dashboardSidebar, 8, 9, 19–21
dropdownMenu, 6, 10, 11, 13
dropdownMenuOutput, 11, 12, 13, 15, 20

dashboardBody, 5, 8
dashboardHeader, 6, 8, 11, 14, 24
dashboardPage, 6, 8
dashboardSidebar, 8, 9, 19–21
dropdownMenu, 6, 10, 11, 13
dropdownMenuOutput, 11, 12, 13, 15, 20

HTML, 14, 15, 17

icon, 12–14, 19, 20, 26
infoBox, 3, 11, 17, 22, 26
infoBoxOutput (valueBoxOutput), 26

menuItem, 12, 13, 23
menuItem (sidebarMenu), 18
menuItemOutput, 11, 12, 13, 15, 20
menuOutput, 11, 12, 13, 15, 20

menuItem (sidebarMenu), 18
menuItemOutput, 11, 12, 13, 15, 20
menuOutput, 11, 12, 13, 15, 20

notificationItem, 10, 13, 14, 24
renderDropdownMenu, 14
renderInfoBox (renderValueBox), 17
renderMenu, 11–13, 15, 19, 20
renderValueBox, 16, 26
shinydashboard, 18
shinydashboard-package (shinydashboard), 18
sidebarMenu, 9, 13, 18, 20, 21, 23
sidebarMenuOutput, 11–13, 15, 19, 23
sidebarSearchForm, 9, 19, 20, 21
sidebarUserPanel, 19, 20, 21
tabBox, 3, 12, 21, 26
tabItem, 23, 24
tabItems, 5, 6, 23, 24

tabPanel, 19, 21
taskItem, 10, 13, 14, 24

uiOutput, 15
updateTabItems, 24
updateTabsetPanel, 24
validColors, 2, 12, 19, 24, 26
validStatuses, 2, 10, 14
valueBox, 3, 6, 12, 17, 22, 25, 26
valueBoxOutput, 17, 26