

# Package: nextGenShinyApps (via r-universe)

August 31, 2024

**Type** Package

**Title** Craft Exceptional 'R Shiny' Applications and Dashboards with Novel Responsive Tools

**Version** 2.0.1

**Maintainer** Obinna Obianom <idonshayo@gmail.com>

**Description** Nove responsive tools for designing and developing 'Shiny' dashboards and applications. The scripts and style sheets are based on 'jQuery' <<https://jquery.com/>> and 'Bootstrap' <<https://getbootstrap.com/>>.

**License** MIT + file LICENSE

**URL** <https://nextgenshinyapps.obianom.com>

**BugReports** <https://github.com/oobianom/nextGenShinyApps>

**Depends** R (>= 3.4)

**Imports** utils, shiny, htmltools, quickcode

**Suggests** rmarkdown, knitr, r2dictionary, r2resize, r2social

**Encoding** UTF-8

**VignetteBuilder** knitr

**Language** en-US

**LazyData** false

**RoxygenNote** 7.2.3

**Repository** <https://oobianom.r-universe.dev>

**RemoteUrl** <https://github.com/oobianom/nextgenshinyapps>

**RemoteRef** HEAD

**RemoteSha** addfb1cc8105e5cd89f67f44c70f14eef4a85f45

## Contents

accordion	2
accordionItem	4
actionButton	5
alert	6
altPanel	7
card	8
checkboxInput	11
cssjsinclude	12
dashboardBody	12
flexBox	13
fluidPage	14
load.example	15
mainPanel	16
masterButton	17
modal.header	18
modalDialog	19
nav	20
rand.num	21
row	22
setup.toolbar.buttons	22
setup.toolbar.menu	23
sidebarLayout	23
sidebarPanel	24
sortablegrid	25
spinner	26
submitButton	27
tabPanel	28
tabsetPanel	29
template.loc	30
textAreaInput	31
textInput	32
titlePanel	34
wrapper	35
<b>Index</b>	<b>36</b>

---

accordion	<i>Generate an accordion</i>
-----------	------------------------------

---

### Description

Wrap one or more accordion items into a container

### Usage

```
accordion(..., id, style = c("default", "1", "2", "3", "4"), uncollapsed = 1)
```

**Arguments**

...	The accordionItem elements to include within the body of the particular accordion
id	The identification of the accordion
style	Style of the accordion, use "default", "1", "2", "3", "4"
uncollapsed	Indicate by number which accordionItem should not be collapsed

**Value**

HTML of a container with a class called accordion

**Examples**

```
if(interactive()){  
  
# Example 1  
  
library(shiny)  
library(nextGenShinyApps)  
  
shiny::shinyApp(  
  ui = fluidPage(  
    style = "8",  
    custom.bg.color = "#d9d9d9",  
    sidebar = NULL,  
    header = titlePanel(left="Sample Ex1"),  
    accordion(  
      id = "accordion5",  
      style = "4",  
      accordionItem(  
        title = "Accordion 1",  
        "Massa sed elementum sus"  
      ),  
      accordionItem(  
        title = "Accordion 2",  
        "Auctor neque etiam non."  
      )  
    )  
  ),  
  server = function(input, output) {  
  }  
)  
  
# Example 2  
  
library(shiny)  
library(nextGenShinyApps)  
  
shiny::shinyApp(  
  ui = fluidPage(  
    style = "8",  
    custom.bg.color = "#d9d9d9",  
    sidebar = NULL,  
    header = titlePanel(left="Sample Ex1"),  
    accordion(  
      id = "accordion5",  
      style = "4",  
      accordionItem(  
        title = "Accordion 1",  
        "Massa sed elementum sus"  
      ),  
      accordionItem(  
        title = "Accordion 2",  
        "Auctor neque etiam non."  
      )  
    )  
  ),  
  server = function(input, output) {  
  }  
)
```

```

    style = "8",
    custom.bg.color = "#f7c4bb",
    sidebar = NULL,
    header = titlePanel(left="Sample Ex2"),
    accordion(
      id = "accordion5",
      style = "1",
      accordionItem(
        title = "Accordion A",
        icon = shiny::icon("trash"),
        "Massa sed elementum sus"
      ),
      accordionItem(
        title = "Accordion B",
        icon = shiny::icon("edit"),
        "Auctor neque etiam non."
      )
    )
  ),
  server = function(input, output) {
  }
)
}

```

---

 accordionItem

*Generate an accordion item*


---

## Description

Embed an accordion item within an accordion

## Usage

```

accordionItem(
  ...,
  title = "A title",
  status = c("default", "primary", "secondary", "info", "success", "danger", "warning"),
  icon = NULL
)

```

## Arguments

...	The elements to include within the body of the particular accordion
title	The title of the accordion item

status	Set the header background using either of "default", "primary", "secondary", "info", "success", "danger", "warning"
icon	Include an icon to the left of the title for the accordion item

**Value**

A list of properties for an accordion item

**Examples**

```
if (interactive()) {
  accordionItem(
    title = "Accordion 2",
    icon = shiny::icon("cog"),
    "Auctor neque etiam non."
  )
}
```

---

actionButton	<i>Create a button</i>
--------------	------------------------

---

**Description**

Upgrade to the actionButton in 'Shiny' package

**Usage**

```
actionButton(
  inputId,
  label,
  icon = NULL,
  width = NULL,
  ...,
  size = c("m", "xs", "s", "l", "xl"),
  style = c("default", "pill", "round", "clean"),
  bg.type = c("default", "primary", "secondary", "info", "success", "danger", "warning"),
  outline = FALSE
)
```

**Arguments**

inputId	Input identification
label	Input label
icon	Choice of button icon
width	Width of the bottom
...	other elements or attributes for the button

size	Size of the button, choices include "m", "xs", "s", "l", "xl"
style	Style of the button, choices include "default", "pill", "round", "clean"
bg.type	Color of the button, choices include "default", "primary", "secondary", "info", "success", "danger", "warning"
outline	Use an outline styling, TRUE or FALSE

**Value**

HTML of the buttons to insert into a page

**Examples**

```
if (interactive()) {
  shiny::div(actionButton("button",
    "Action button with primary color",
    icon = shiny::icon("folder"), bg.type = "primary"
  ))
  shiny::div(actionButton("button",
    "Action button with primary color",
    icon = shiny::icon("file"),
    bg.type = "danger", outline = TRUE
  ))
}
```

---

alert

*Create an alert*

---

**Description**

Create an alert with various styles

**Usage**

```
alert(
  ...,
  type = c("default", "standard"),
  close = FALSE,
  color = c("none", "primary", "secondary", "info", "success", "danger", "warning"),
  outline = FALSE,
  icon = NULL
)
```

**Arguments**

...	Content of the alert
type	Choose a style for the alert, choices include "default", "standard"
close	Allow alert to be closable, TRUE or FALSE
color	Color of the alert, choose between "none", "primary", "secondary", "info", "success", "danger", "warning"
outline	Include an outline and exclude background, TRUE or FALSE
icon	Include an icon to the left of the content

**Value**

HTML of an alert box to be inserted within a page

**Examples**

```
if (interactive()) {
  card(
    header = FALSE,
    shiny::h2("Standard alert (closeable)"),
    alert("EX1", type = "standard",
      color = "primary"),
    alert("EX2", type = "standard",
      color = "secondary"),
    alert("EX3", type = "standard",
      color = "secondary", outline = TRUE),
    alert("EX4", type = "standard",
      color = "danger", outline = TRUE, close = TRUE),
    alert("EX5", type = "standard",
      close = TRUE),
    alert("EX6", type = "standard",
      color = "primary", icon = shiny::icon("info"))
  )
}
```

---

altPanel

*New sidebar panel to display content*


---

**Description**

Customizable sidebar panel for inclusion of various UI elements

**Usage**

```
altPanel(..., width = 4, border = FALSE, shadow = FALSE)
```

**Arguments**

...	List of content
width	Width of the sidebar panel
border	Should border be declared for the panel
shadow	Should a shadow be added to the panel

**Value**

Creates an alternate container for displaying contents

**Note**

For more information on the features of the sidebar panel, look through the Github examples

**Examples**

```
altPanel('content 2')
```

---

card

*Generate a flexible and extensible content container*

---

**Description**

Widely used Bootstrap feature with improvements to allow collapse, minimize and closing

**Usage**

```
card(  
  ...,  
  title = "Standard Card",  
  collapsed = FALSE,  
  bg.fade = TRUE,  
  width = 12,  
  header.bg = letters[1:13],  
  alert.text = NULL,  
  alert.bg = c("primary", "warning", "secondary", "info", "success", "danger"),  
  toolbar = NULL,  
  header = TRUE,  
  draggable = TRUE,  
  id = NULL  
)
```



**Arguments**

...	The elements to include within the body of the card
title	The text to display in the header title
collapsed	If TRUE, the card is collapsed. The default is FALSE
bg.fade	If TRUE, the background will be faded if a background exists
width	Select a width from 1 to 12 to indicate the size of the card
header.bg	Header background color style, choices 1 to 11
alert.text	Enter text for the alert portion. Leave as NULL to exclude the alert
alert.bg	Indicate the type of alert to include, choices are "primary", "warning", "secondary", "info", "success", "danger"
toolbar	The default is NULL, which means all toolbar will be displayed use this to set what toolbar to show.
header	If FALSE, the header will be excluded
draggable	If FALSE, the card will not be draggable
id	unique card id

**Value**

HTML code of the container with a class called card that holds the items

**Note**

For more information on the features of the card, visit the examples section of the help documentation

**Examples**

```
# Example 1
if (interactive()) {
  library(shiny)
  library(nextGenShinyApps)

  shiny::shinyApp(
    ui = fluidPage(
      style = "4",
      custom.bg.color = "lightblue",
      sidebar = NULL,
      header = titlePanel(left="Card Ex2"),
      wrapper(
        altPanel(
          card(
            title = "Standard card",
            collapsed = TRUE,
            alert.text = "An alert2 for the content",
            alert.bg = "warning",
            toolbar = list(collapse = TRUE,
                          maximize = TRUE,
```

```

        close = FALSE,
        menu = TRUE),
      shiny::h3("Sample text"),
      "Lorem ipsum dolor sit a"
    )),
  mainPanel(
    card(
      title = "Standard card 2",
      shiny::h1("Sample text"),
      "Lorem ipsum dolor sit a"
    )
  )
),
server = function(input, output) {
}
}

# Example 2
if (interactive()) {
  library(shiny)
  library(nextGenShinyApps)

  shiny::shinyApp(
    ui = fluidPage(
      style = "8",
      custom.bg.color = "#d9d9d9",
      sidebar = NULL,
      header = titlePanel(left="Card Ex1"),
      wrapper(
        altPanel(width = 12,
          card(
            title = "Standard card",
            alert.text = "An alert1 for the content",
            shiny::h3("Sample text"),
            "Lorem ipsum dolor sit a"
          ),
          card(
            title = "Standard card",
            collapsed = TRUE,
            alert.text = "An alert2 for the content",
            alert.bg = "warning",
            toolbar = list(collapse = TRUE,
                          maximize = TRUE,
                          close = FALSE,
                          menu = TRUE),
            shiny::h3("Sample text"),
            "Lorem ipsum dolor sit a"
          )
        )
      ),
    server = function(input, output) {
}
}

```

```
    )  
}
```

---

`checkboxInput`*Create an advanced checkbox input*

---

**Description**

Modifications to checkboxinput to allow added styles

**Usage**

```
checkboxInput(inputId, label, value = FALSE, width = NULL, inline = FALSE)
```

**Arguments**

<code>inputId</code>	The identification name
<code>label</code>	The label for the input
<code>value</code>	The current value of the input
<code>width</code>	width of the text input
<code>inline</code>	make inline or block format

**Value**

HTML elements of a checkbox

**Note**

For more information on the features of the form, visit the examples section of the help documentation

**Examples**

```
checkboxInput("somevalue", "Some value", FALSE)  
checkboxInput("somevalue", "Some value", value = FALSE, inline = TRUE)  
checkboxInput("somevalue", "Some value", FALSE)
```

cssjsinclude                    *Include stylesheets and scripts*

---

**Description**

Use the package scripts and stylesheets in a page

**Usage**

```
cssjsinclude(template, color)
```

**Arguments**

template	The template type
color	The numeric style of template

**Value**

A list of files to be inserted in the header of a page

**Examples**

```
cssjsinclude('core', '3')
```

---

dashboardBody                    *Create the body section of the application*

---

**Description**

Create a simple body containing a header and a content for the main body

**Usage**

```
dashboardBody(header, ...)
```

**Arguments**

header	OPTIONAL. Items to display in the header section (use the titlePanel() function to set this property).
...	The elements to include within the body of the modal

**Value**

An HTML of the body of the page

**Note**

Endeavor to use as standalone and not within the fluidPage, as this function it already called within fluidPage

**Examples**

```
if (interactive()) {  
  dashboardBody(  
    header = titlePanel(  
      left = "Sample nextGenShinyApps Title",  
      right = shiny::icon("user")  
    ),  
    "sample text for main body"  
  )  
}
```

---

flexBox

*Container to fit all containers horizontally*

---

**Description**

Fit containers horizontally

**Usage**

```
flexBox(..., border = FALSE, shadow = FALSE)
```

**Arguments**

...	List of content
border	Should border be declared for the panel
shadow	Should a shadow be added to the panel

**Value**

Creates a a box to fit all boxes within

**Examples**

```
flexBox(shiny::div("Div 1"),shiny::div("Div 2"),shiny::div("Div 1"))
```

---

fluidPage	<i>Generate a container for the application</i>
-----------	---

---

### Description

An upgrade to the fluidPage function available in the 'Shiny' package

### Usage

```
fluidPage(
  ...,
  id = NULL,
  header = NULL,
  sidebar = NULL,
  class = NULL,
  style = rand.sc13,
  custom.bg.color = NULL,
  modal.header.links = NULL
)
```

### Arguments

...	The elements to include within the body of the page
id	OPTIONAL. Identification tag of the container
header	OPTIONAL. Items to display in the header section (use the titlePanel() function to set this property).
sidebar	OPTIONAL. Items to display in the sidebar section (use the sidebarPanel() function to set this property).
class	OPTIONAL. Class name for the container
style	OPTIONAL. Various unique styles choices from 1 - 12
custom.bg.color	OPTIONAL. Choice to change the background color of the container. Use HEX values such as #FFFFFF or RGB code such as rgb(255,255,255) or simple color name such as 'lightblue'
modal.header.links	OPTIONAL. One or more list containing links that appear when the app title is clicked. To remove, set to NULL

### Value

A rendered HTML of the container holder of the application items

### Note

This global layout can be applied to a variety of shiny app and dashboard, with or without a sidebar. See the example below.

**Examples**

```

if (interactive()) {
  library(shiny)
  library(nextGenShinyApps)

  shiny::shinyApp(
    ui = fluidPage(
      style = "6",
      header = titlePanel(left = "Sample App Title",
        right = "Image/logo"),
      sidebar = sidebarPanel(
        title = "myApp",
        "Sample sidebar content"
      ),
      "Sample body content"
    ),
    server = function(input, output) {}
  )
}

```

---

load.example

*Load examples for the package*


---

**Description**

Example shiny applications spanning aspects of the package

**Usage**

```

load.example(
  example = c("Plain", "noSideBar", "Plotly", "Rhansometable", "Card", "Tab", "Modal",
    "Form", "Button", "Spinner", "Alert", "Accordion")
)

```

**Arguments**

example	choose the example to show - "Plain","noSideBar","Plotly","Rhansometable", "Card","Tab","Modal","Form","Button","Spinner","Alert","Accordion"
---------	--

**Value**

A rendered HTML of the user specified example file

**Options for example**

"Plain", "noSideBar", "Plotly", "Rhansometable", "Card", "Tab", "Modal", "Form", "Button", "Spinner", "Alert", "Accordion"

## Examples

```
if (interactive()) {  
  load.example(example = "Card")  
}
```

---

mainPanel	<i>Main panel to display content</i>
-----------	--------------------------------------

---

## Description

Customizable main panel for inclusion of various UI elements

## Usage

```
mainPanel(..., width = 8, border = FALSE, shadow = FALSE)
```

## Arguments

...	List of content
width	Width of the main panel
border	Should border be declared for the panel
shadow	Should a shadow be added to the panel

## Value

Creates a container for displaying contents

## Note

For more information on the features of the main panel, look through the Github examples

## Examples

```
mainPanel('content 1')
```



---

masterButton	<i>Create a master button</i>
--------------	-------------------------------

---

## Description

A master button creator

## Usage

```
masterButton(  
    inputId,  
    text = "Text",  
    icon = NULL,  
    width = NULL,  
    size = c("m", "xs", "s", "l", "xl"),  
    style = c("default", "pill", "round", "clean"),  
    bg.type = c("default", "primary", "secondary", "info", "success", "danger", "warning"),  
    outline = FALSE,  
    extraClass = NULL,  
    ...  
)
```

## Arguments

inputId	input id
text	Button text
icon	Choice of button icon
width	Width of the bottom
size	Size of the button, choices include "m","xs", "s", "l", "xl"
style	Style of the button, choices include "default", "pill", "round", "clean"
bg.type	Color of the button, choices include "default", "primary", "secondary", "info", "success", "danger", "warning"
outline	Use an outline styling, TRUE or FALSE
extraClass	other class names to add to the button attribute
...	Other elements to add to the button

## Value

HTML of the buttons to insert into a page

**Examples**

```

if (interactive()) {
  card(
    shiny::h2("Master buttons with various styles"),
    header = FALSE,
    shiny::div(masterButton("Submit button with primary color",
      icon = shiny::icon("file"), size = "s", bg.type = "primary"
    ))
  )
}

```

---

modal.header

*Create hyperlink modal section that appears ONLY when the sidebar logo is clicked*

---

**Description**

Additional tab section for additional links

**Usage**

```
modal.header(...)
```

**Arguments**

...                   The list of tabs to include

**Value**

An HTML containing elements of links to be inserted in the header of a page

**Examples**

```

if (interactive()) {
  list1 <- list(
    title = "Home", icon = shiny::icon("home"),
    link = "https://google.com"
  )
  list2 <- list(
    title = "Docs", icon = shiny::icon("folder"),
    link = "https://obi.obianom.com"
  )

  modal.header(list(list1, list2))
}

```

---

modalDialog	<i>Generate a modal box</i>
-------------	-----------------------------

---

**Description**

Advanced modal dialog that allows various positioning and transparency

**Usage**

```
modalDialog(  
  ...,  
  title = NULL,  
  footer = modalButton("Dismiss"),  
  size = c("m", "s", "l", "xl"),  
  easyClose = FALSE,  
  fade = TRUE,  
  position = c("centered", "left", "right", "top", "bottom"),  
  transparent = FALSE  
)
```

**Arguments**

...	The elements to include within the body of the modal
title	The text to display in the header title
footer	Footer list of buttons or text
size	The size of the modal, "m", "s", "l", "xl"
easyClose	Allow simple closing, FALSE or TRUE
fade	Allow fading away or fading in, FALSE or TRUE
position	Set the position of the modal. Choices include "centered", "left", "right", "top", "bottom"
transparent	Allow background transparency, FALSE or TRUE

**Value**

An HTML containing elements of a modal box that remains hidden until a button is clicked

**Note**

For more information on the features of the card, visit the examples section of the help documentation

## Examples

```
if (interactive()) {
  library(shiny)
  library(nextGenShinyApps)
  shiny::shinyApp(
    ui = fluidPage(
      style = "8",
      custom.bg.color = "white",
      sidebar = NULL,
      header = NULL,
      shiny::h3("Modal EXAMPLES"),
      shiny::div(actionButton("obianom1", "Show BIG shiny modal on the RIGHT")),
      shiny::br(),
      shiny::div(actionButton("obianom2", "Show SMALL shiny modal on the RIGHT"))
    ),
    server = function(input, output) {
      shiny::observeEvent(input$obianom1, {
        shiny::showModal(modalDialog(
          textInput("dataset", "Enter a data set"),
          shiny::div("Id leo in vitae"),
          size = "1",
          position = "bottom",
        ))
      })
      shiny::observeEvent(input$obianom2, {
        shiny::showModal(modalDialog(
          textInput("dataset", "Enter a data set"),
          shiny::div("Lorem donec massa"),
          size = "1",
          position = "right",
        ))
      })
    }
  )
}
```

---

nav

*Nav tag*

---

## Description

A nav tag for creating HTML navigations

## Usage

```
nav(class, id = NULL, role = NULL, ...)
```

**Arguments**

- class           The class of the navigation container
- id             The identification of the navigation container
- role           The character role of the container on the page
- ...            The content of the container

**Value**

HTML content of a container with type nav

**Examples**

```
nav('sample','id1','sample','some content')
```

---

rand.num	<i>Random number between 1 and 10000</i>
----------	--

---

**Description**

One or more random numbers

**Usage**

```
rand.num(num)
```

**Arguments**

- num            The number of numbers to return

**Value**

One or more numbers

**Examples**

```
rand.num(10)
```

---

row	<i>Generate a row div</i>
-----	---------------------------

---

**Description**

A simple row div

**Usage**

```
row(...)
```

**Arguments**

...           The elements to include within the body of the row

**Value**

An HTML containing elements of a container with class row to be embedded in a page

**Examples**

```
row(shiny::div(width=12,"Hello nextGenShinyApps"))
```

---

setup.toolbar.buttons	<i>Generate toolbar buttons</i>
-----------------------	---------------------------------

---

**Description**

Use within a card to display toolbar

**Usage**

```
setup.toolbar.buttons(...)
```

**Arguments**

...           The list of buttons to display

**Value**

HTML code of a container containing items to be inserted in the toolbar

**Note**

For more information on the features of a toolbar within a card, visit the examples section of the help documentation

**Examples**

```
setup.toolbar.buttons(list(maximize=TRUE,collapse=TRUE,close=TRUE))
```

---

setup.toolbar.menu	<i>Generate toolbar menu</i>
--------------------	------------------------------

---

**Description**

Use within a card to display menu

**Usage**

```
setup.toolbar.menu(...)
```

**Arguments**

... The list declaring whether to show menu

**Value**

HTML code of a container containing menu to be inserted in the toolbar if declared TRUE

**Note**

For more information on the features of a toolbar within a card, visit the examples section of the help documentation

**Examples**

```
setup.toolbar.menu(list(menu=TRUE))
```

---

sidebarLayout	<i>Sidebar layout</i>
---------------	-----------------------

---

**Description**

Container for sidebar, optional

**Usage**

```
sidebarLayout(...)
```

**Arguments**

... content of the sidebar layout

**Value**

An HTML containing elements in a container

**Examples**

```
sidebarLayout("sample text")
```

---

sidebarPanel	<i>Create the sidebar panel</i>
--------------	---------------------------------

---

**Description**

Creates an advanced sidebar panel with colors corresponding to a chosen theme style

**Usage**

```
sidebarPanel(..., title = "TitleApp", footer = FALSE)
```

**Arguments**

...	The elements to include within the body of the sidebar
title	Title of the sidebar content
footer	Whether or not to enable the footer content

**Value**

An HTML containing elements of a sidebar to be embedded in a page

**Note**

This global layout can be applied to a variety of shiny app and dashboard, with or without a sidebar. See the example below.

**Examples**

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

  shiny::shinyApp(
    ui = fluidPage(
      style = "7",
      header = NULL,
      sidebar = sidebarPanel(
        title = "myApp",
        "Sample sidebar contents",
        footer = FALSE
      ),
    ),
```



```
        "Plain content"
    ),
    server = function(input, output) {}
  )
}
```

---

sortablegrid	<i>Generate a sortable grid</i>
--------------	---------------------------------

---

## Description

A grid that holds draggable items

## Usage

```
sortablegrid(..., width = 6, id = NULL)
```

## Arguments

...	The elements to include within the body of the grid
width	The width of the grid
id	unique id of grid

## Value

HTML code of a container that allows items within it to be draggable

## Note

For more information on the features of a sortable grid, visit the examples section of the help documentation

## Examples

```
sortablegrid("item1",width=12)
```

---

`spinner`*Create a spinner*

---

## Description

Create a loading spinner for customization of outputs

## Usage

```
spinner(  
  type = c("ring", "grow", "square", "rect"),  
  size = c("l", "s"),  
  color = c("default", "primary", "secondary", "info", "success", "danger", "warning")  
)
```

## Arguments

<code>type</code>	Choose a style for the spinner using "ring","grow","square","rect"
<code>size</code>	Size of the spinner, "l" for large and "s" for small
<code>color</code>	Color of the spinner, choose between "default", "primary", "secondary", "info", "success", "danger", "warning"

## Value

An HTML containing elements of a spinner to be embedded in a page during loading

## Examples

```
if (interactive()) {  
  library(shiny)  
  library(nextGenShinyApps)  
  
  shiny::shinyApp(  
    ui = fluidPage(  
      style = "3",  
      custom.bg.color = "cyan",  
      sidebar = NULL,  
      header = NULL,  
      card(  
        header = FALSE,  
        shiny::h2("loading spinner"),  
        spinner(type = "rect", size = "s"),  
        spinner(type = "rect", color = "primary"),  
        spinner(type = "grow", color = "secondary"),  
        spinner(type = "ring", color = "success"),  
        spinner(type = "rect", color = "warning"),  
        spinner(type = "square", color = "danger"),  
        spinner(type = "rect", color = "info")  
      )  
    )  
  )  
}
```

```

    )
  },
  server = function(input, output) {
  }
)
}

```

---

submitButton

*Create a submit button*


---

### Description

Upgrade to the submitButton in 'Shiny' package

### Usage

```

submitButton(
  inputId,
  text = "Apply Changes",
  icon = NULL,
  width = NULL,
  size = c("m", "xs", "s", "l", "xl"),
  style = c("default", "pill", "round", "clean"),
  bg.type = c("default", "primary", "secondary", "info", "success", "danger", "warning"),
  outline = FALSE
)

```

### Arguments

inputId	input id
text	Button text
icon	Choice of button icon
width	Width of the bottom
size	Size of the button, choices include "m","xs", "s", "l", "xl"
style	Style of the button, choices include "default", "pill", "round", "clean"
bg.type	Color of the button, choices include "default", "primary", "secondary", "info", "success", "danger", "warning"
outline	Use an outline styling, TRUE or FALSE

### Value

HTML of the submit buttons to insert into a page

## Examples

```
if (interactive()) {
  card(
    shiny::h2("Submit buttons with various styles"),
    header = FALSE,
    shiny::div(submitButton("Submit button with primary color",
      icon = shiny::icon("file"), size = "s", bg.type = "primary"
    )),
    shiny::div(submitButton("Secondary color",
      icon = shiny::icon("folder"), bg.type = "secondary"
    )),
    shiny::div(submitButton("Success color",
      icon = shiny::icon("filter"), bg.type = "success"
    )),
    shiny::div(submitButton("Warning color",
      icon = shiny::icon("grid"), bg.type = "warning"
    )),
    shiny::div(submitButton("Danger color",
      icon = shiny::icon("check"), bg.type = "danger"
    )),
    shiny::div(submitButton("Info color",
      icon = shiny::icon("trash"), bg.type = "info"
    ))
  )
}
```

---

tabPanel

*Create a tab panel item*

---

## Description

Create a tab panel item that is enclosed by a `tabsetPanel`

## Usage

```
tabPanel(title, ...)
```

## Arguments

<code>title</code>	title of the tab
<code>...</code>	content of the tab

## Value

An list containing the title and content of a tab

## Examples

```
if (interactive()) {  
  tabPanel("Summary", "Convallis aesus.")  
  tabPanel("Summary", "nextGenShinyAppss.")  
}
```

---

tabsetPanel

*Create an advanced tabset*

---

## Description

Advanced tabset panel with styles and features

## Usage

```
tabsetPanel(  
  ...,  
  type = c("default", "pills", "clean", "jPills", "justified"),  
  border = TRUE,  
  justified = FALSE,  
  position = c("left", "end", "center")  
)
```

## Arguments

...	Content of the tabset, created using the tabPanel for each individual item
type	Type of tabset to create, choices include "default","pills","clean","jPills","justified"
border	Include a board for the tabset, TRUE or FALSE
justified	Justify tab headers, TRUE or FALSE
position	position of the tabs, choices include "left","end","center"

## Value

An HTML containing elements of a tabset to be embedded in a page

## Note

Many examples exist for the tabset, fid them using the package load.example function

## Examples

```
if (interactive()) {
  library(shiny)
  library(nextGenShinyApps)
  tab2 <- tabPanel("Summary", "SAMPLE nunc.")
  tab3 <- tabPanel("Tab 3", "aoreet sit amet.")
  tab4 <- tabPanel("Tab 4", "Vulputate pulvinar")

  shiny::shinyApp(
    ui = fluidPage(
      style = "8",
      custom.bg.color = "rgb(110,134,032)",
      sidebar = NULL,
      header = NULL,
      tabsetPanel(
        tab2,
        tab3,
        type = "pills",
        justified = TRUE
      )
    ),
    server = function(input, output) {
    }
  )
}
```

---

template.loc

*Template location full text*

---

## Description

Fetch the location of the scripts

## Usage

```
template.loc(template = "core")
```

## Arguments

template      The type of template to fetch

## Value

A path for the location of the package

**Examples**

```
template.loc('core')
```

---

textAreaInput

*Create an advanced text area input*


---

**Description**

Modifications to 'textAreaInput' to allow added styles

**Usage**

```
textAreaInput(
  inputId,
  label,
  value = "",
  width = NULL,
  height = NULL,
  cols = NULL,
  rows = NULL,
  placeholder = NULL,
  resize = c("both", "none", "vertical", "horizontal"),
  style = c("default", "pill", "round", "clean"),
  border.type = c("none", "primary", "secondary", "info", "success", "danger", "warning")
)
```

**Arguments**

inputId	The identification name
label	The label for the input
value	The current value of the input
width	width of the text input
height	height of the text input
cols	col of text to display
rows	row of text to display
placeholder	A placeholder text
resize	Make inout resizable, with choices "both", "none", "vertical", "horizontal"
style	Style to adapt, options include "default", "pill", "round", "clean"
border.type	Add a border coloring using either of "none", "primary", "secondary", "info", "success", "danger", "warning"

**Value**

HTML element of a textAreaInput

**Note**

For more information on the features of the form, visit the examples section of the help documentation

**Examples**

```
textInput("caption",
  "Sample Text area input",
  "Data Summary",
  width = "1000px", border.type = "success"
)
```

---

 textInput

*Create an advanced text input*


---

**Description**

Modifications to textInput to allow added functionality and styles

**Usage**

```
textInput(
  inputId,
  label,
  value = "",
  width = NULL,
  placeholder = NULL,
  size = c("m", "s", "l", "xl"),
  style = c("default", "pill", "round", "clean"),
  border.type = c("none", "primary", "secondary", "info", "success", "danger", "warning"),
  prepend = NULL,
  append = NULL,
  disabled = FALSE
)
```

**Arguments**

inputId	The identification name
label	The label for the input
value	The current value of the input
width	width of the text input
placeholder	A placeholder text
size	The size of the input, "m", "s", "l", "xl"
style	Style to adapt, options include "default", "pill", "round", "clean"



border.type	Add a border coloring using either of "none", "primary","secondary", "info", "success", "danger", "warning"
prepend	Add a prepended text or icon
append	Add an appended text or icon
disabled	Boolean. If textInput should be disabled

**Value**

A HTML with modifications to th style information

**Note**

For more information on the features of the form, visit the examples section of the help documentation

**Examples**

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

  shiny::shinyApp(
    ui = fluidPage(
      style = "8",
      custom.bg.color = "white",
      sidebar = NULL,
      header = NULL,
      card(
        header = FALSE,
        tags$h3("Text input"),
        textInput("caption", "Basic"),
        textInput("caption", "Basic", style = "clean"),
        textInput("caption", "Border - primary",
          "Enter sample text",
          prepend = "@", border.type = "info"
        ),
        textInput("caption", "Border - primary",
          "Enter sample text",
          prepend = shiny::icon("lock")
        ),
        textInput("caption", "Border - primary",
          "Enter sample text",
          append = "%"
        ),
        textInput("caption", "Border - primary",
          "Enter sample text",
          prepend = shiny::icon("chart"),
          append = ".00"
        )
      )
    )
  ),
)
```

```
server = function(input, output) {  
  }  
}  
}
```

---

titlePanel	<i>Title panel for the header of the application</i>
------------	--

---

### Description

Used to embed the header within the body of the application

### Usage

```
titlePanel(left = "Sample Tile", right = NULL, link = "#")
```

### Arguments

left	The title text for the header
right	Content to include on the top right corner
link	Hyperlink to navigate when clicked

### Value

An HTML containing elements to insert in a title

### Examples

```
if (interactive()) {  
  titlePanel(  
    left = "Sample App Title",  
    right = shiny::div("Image/logo", shiny::icon("trash"))  
  )  
}
```

---

wrapper	<i>A wrapper for panels</i>
---------	-----------------------------

---

**Description**

Create a wrapper div for pannels

**Usage**

```
wrapper(  
  ...,  
  bg = c("default", "primary", "secondary", "warning", "info", "danger", "success")  
)
```

**Arguments**

...	div contents
bg	background color of the wrapper

**Value**

a container for other containers

**Examples**

```
wrapper(altPanel("hello"),mainPanel("test"))  
wrapper(mainPanel("hello"),shiny::column(width=2,"test"))
```

# Index

accordion, 2  
accordionItem, 4  
actionButton, 5  
alert, 6  
altPanel, 7  
  
card, 8  
checkboxInput, 11  
cssjsinclude, 12  
  
dashboardBody, 12  
  
flexBox, 13  
fluidPage, 14  
  
load.example, 15  
  
mainPanel, 16  
masterButton, 17  
modal.header, 18  
modalDialog, 19  
  
nav, 20  
  
rand.num, 21  
row, 22  
  
setup.toolbar.buttons, 22  
setup.toolbar.menu, 23  
sidebarLayout, 23  
sidebarPanel, 24  
sortablegrid, 25  
spinner, 26  
submitButton, 27  
  
tabPanel, 28  
tabsetPanel, 29  
template.loc, 30  
textAreaInput, 31  
textInput, 32  
titlePanel, 34  
  
wrapper, 35