

Package ‘shinythemes’

January 25, 2021

Title Themes for Shiny

Version 1.2.0

Description Themes for use with Shiny. Includes several Bootstrap themes from <<https://bootswatch.com/>>, which are packaged for use with Shiny applications.

Depends R (>= 3.0.0)

Imports shiny (>= 0.11)

URL <https://rstudio.github.io/shinythemes/>

License GPL-3 | file LICENSE

RoxygenNote 7.1.1

NeedsCompilation no

Author Winston Chang [aut, cre],
RStudio [cph],
Thomas Park [ctb, cph] (Bootswatch themes),
Lukasz Dziedzic [ctb, cph] (Lato font),
Nathan Willis [ctb, cph] (News Cycle font),
Google Corporation [ctb, cph] (Open Sans and Roboto fonts),
Matt McInerney [ctb, cph] (Raleway font),
Adobe Systems Incorporated [ctb, cph] (Source Sans Pro font),
Canonical Ltd [ctb, cph] (Ubuntu font)

Maintainer Winston Chang <winston@rstudio.com>

Repository CRAN

Date/Publication 2021-01-25 22:00:02 UTC

R topics documented:

shinytheme	2
shinythemes	2
themeSelector	5

Index	7
--------------	----------

shinytheme *Return the URL for a Shiny theme*

Description

The result of this function should be used as the theme argument for [bootstrapPage](#), [fluidPage](#), [navbarPage](#), or [fixedPage](#).

Usage

```
shinytheme(theme = NULL)
```

Arguments

theme Name of a theme.

See Also

The main [shinythemes](#) page for information about available themes and more detailed examples.

Examples

```
## Not run:
shinyApp(
  ui = fluidPage(theme = shinytheme("united"),
    ...
  ),
  server = function(input, output) { }
)

## End(Not run)
```

shinythemes *Themes for Shiny*

Description

This package contains Bootstrap themes from <https://bootswatch.com/>, which are packaged for use with Shiny applications. The themes included are:

Details

- [cerulean](#)
- [cosmo](#)
- [cyborg](#)
- [darkly](#)
- [flatly](#)
- [journal](#)
- [lumen](#)
- [paper](#)
- [readable](#)
- [sandstone](#)
- [simplex](#)
- [slate](#)
- [spacelab](#)
- [superhero](#)
- [united](#)
- [yeti](#)

To use the themes, use the theme argument to [bootstrapPage](#), [fluidPage](#), [navbarPage](#), or [fixedPage](#). The value should be `shinytheme("cerulean")`, where the theme name takes the place of "cerulean".

Examples

```
## Not run:
library(shiny)
library(shinythemes)

# A very basic navbar page with different themes
shinyApp(
  ui = navbarPage("Default theme",
    tabPanel("Plot", "Plot tab contents..."),
    navbarMenu("More",
      tabPanel("Summary", "Summary tab contents..."),
      tabPanel("Table", "Table tab contents...")
    )
  ),
  server = function(input, output) { }
)

shinyApp(
  ui = navbarPage("United",
    theme = shinytheme("united"),
    tabPanel("Plot", "Plot tab contents..."),
    navbarMenu("More",
      tabPanel("Summary", "Summary tab contents..."),
      tabPanel("Table", "Table tab contents...")
    )
  )
)
```

```

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

shinyApp(
  ui = navbarPage("Cerulean",
    theme = shinytheme("cerulean"),
    tabPanel("Plot", "Plot tab contents..."),
    navbarMenu("More",
      tabPanel("Summary", "Summary tab contents..."),
      tabPanel("Table", "Table tab contents...")
    )
  ),
  server = function(input, output) { }
)

# A more complicated app with the flatly theme
shinyApp(
  ui = fluidPage(
    theme = shinytheme("flatly"),
    titlePanel("Tabsets"),
    sidebarLayout(
      sidebarPanel(
        radioButtons("dist", "Distribution type:",
          c("Normal" = "norm",
            "Uniform" = "unif",
            "Log-normal" = "lnorm",
            "Exponential" = "exp")),
        br(),
        sliderInput("n", "Number of observations:",
          value = 500, min = 1, max = 1000)
      ),
      mainPanel(
        tabsetPanel(type = "tabs",
          tabPanel("Plot", plotOutput("plot")),
          tabPanel("Summary", verbatimTextOutput("summary")),
          tabPanel("Table", tableOutput("table"))
        )
      )
    ),
  server = function(input, output) {
    data <- reactive({
      dist <- switch(input$dist,
        norm = rnorm,
        unif = runif,
        lnorm = rlnorm,
        exp = rexp,
        rnorm)
      dist(input$n)
    })
  }
)

```

```
output$plot <- renderPlot({
  dist <- input$dist
  n <- input$n
  hist(data(), main=paste('r', dist, '(', n, ')', sep=''))
})

output$summary <- renderPrint({
  summary(data())
})

output$table <- renderTable({
  data.frame(x=data())
})
}
)

## End(Not run)
```

themeSelector

Add a theme selector widget in a floating panel

Description

This adds a widget for selecting the theme, in a floating panel. It is meant for use during the development phase of a Shiny application.

Usage

```
themeSelector()
```

Details

This can be inserted anywhere inside of the application, although if it is put inside a tab, it will be visible only when that tab is showing. For it to show at all times, it must be used outside a tab.

Examples

```
if (interactive()) {
# themeSelector can be inserted anywhere in the app.
shinyApp(
  ui = fluidPage(
    shinythemes::themeSelector(),
    sidebarPanel(
      textInput("txt", "Text input:", "text here"),
      sliderInput("slider", "Slider input:", 1, 100, 30),
      actionButton("action", "Button"),
      actionButton("action2", "Button2", class = "btn-primary")
    ),
  ),
)
```

```

    mainPanel(
      tabsetPanel(
        tabPanel("Tab 1"),
        tabPanel("Tab 2")
      )
    ),
    server = function(input, output) {}
  )

```

If this is used with a navbarPage() or other type of page where there is not a good place to put it where it is outside of all tabs, you can wrap the entire page in tagList() and make the themeSelector a sibling of the page.

```

shinyApp(
  ui = tagList(
    shinythemes::themeSelector(),
    navbarPage(
      "Theme test",
      tabPanel("Navbar 1",
        sidebarPanel(
          textInput("txt", "Text input:", "text here"),
          sliderInput("slider", "Slider input:", 1, 100, 30),
          actionButton("action", "Button"),
          actionButton("action2", "Button2", class = "btn-primary")
        ),
        mainPanel(
          tabsetPanel(
            tabPanel("Tab 1"),
            tabPanel("Tab 2")
          )
        )
      ),
      tabPanel("Navbar 2")
    )
  ),
  server = function(input, output) {}
)
}

```

Index

`bootstrapPage`, [2](#), [3](#)

`fixedPage`, [2](#), [3](#)

`fluidPage`, [2](#), [3](#)

`navbarPage`, [2](#), [3](#)

`shinytheme`, [2](#)

`shinythemes`, [2](#), [2](#)

`themeSelector`, [5](#)